

#2

Docket No. 826.1721

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:)
Hironori YAHAGI)
Serial No.: To be assigned) Group Art Unit: Unassigned
Filed: March 27, 2001) Examiner: Unassigned
For: APPARATUS CONVERTING)
A STRUCTURED DOCUMENT)
HAVING A HIERARCHY)

1c903 U.S. PRO
09/819729
03/29/01

**SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN
APPLICATION IN ACCORDANCE
WITH THE REQUIREMENTS OF 37 C.F.R. §1.55**

*Assistant Commissioner for Patents
Washington, D.C. 20231*

Sir:

In accordance with the provisions of 37 C.F.R. §1.55, the applicant submits herewith a certified copy of the following foreign application:

Japanese Patent Application No. 2000-296161
Filed: September 28, 2000.

It is respectfully requested that the applicant be given the benefit of the foreign filing date as evidenced by the certified papers attached hereto, in accordance with the requirements of 35 U.S.C. §119.

Respectfully submitted,

STAAS & HALSEY LLP

Date: March 27, 2001

By: _____

James D. Halsey, Jr.
Registration No. 22,729

700 Eleventh Street, N.W.
Suite 500
Washington, D.C. 20001
(202) 434-1500

PATENT OFFICE
JAPANESE GOVERNMENT



This is to certify that the annexed is a true copy of the
following application as filed with this Office.

Date of Application: September 28, 2000

Application Number: Patent Application No. 2000-296161

Applicant(s): FUJITSU LIMITED

December 22, 2000

Commissioner,
Patent Office Kozo OIKAWA

Certificate No. 2000-3105882

日 本 国 特 許 庁

PATENT OFFICE
JAPANESE GOVERNMENT

JP903 U.S. PTO
09/819729
03/29/01

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application:

2000年 9月28日

出 願 番 号
Application Number:

特願2000-296161

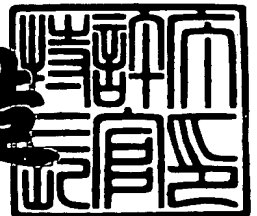
出 願 人
Applicant (s):

富士通株式会社

2000年12月22日

特許庁長官
Commissioner,
Patent Office

及 川 耕 造



出証番号 出証特2000-3105882

【書類名】 特許願

【整理番号】 0051425

【提出日】 平成12年 9月28日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 12/00
G06F 17/30

【発明の名称】 階層構造の構造化文書を変換する装置

【請求項の数】 5

【発明者】

【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

【氏名】 矢作 裕紀

【特許出願人】

【識別番号】 000005223

【氏名又は名称】 富士通株式会社

【代理人】

【識別番号】 100074099

【住所又は居所】 東京都千代田区二番町8番地20 二番町ビル3F

【弁理士】

【氏名又は名称】 大菅 義之

【電話番号】 03-3238-0031

【選任した代理人】

【識別番号】 100067987

【住所又は居所】 神奈川県横浜市鶴見区北寺尾7-25-28-503

【弁理士】

【氏名又は名称】 久木元 彰

【電話番号】 045-573-3683

【手数料の表示】

【予納台帳番号】 012542

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9705047

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 階層構造の構造化文書を変換する装置

【特許請求の範囲】

【請求項 1】 階層構造の要素の集合で記述され、それぞれが 1 つ以上の要素を含む複数のレコードから成る構造化文書の情報を入力する文書入力手段と、

前記構造化文書の 2 つ以上のレコード間で、相対的に同じ位置にある要素の内容を接合して、新しい要素を生成する接合手段と、

前記新しい要素を含み、前記 2 つ以上のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成する生成手段と、

前記 2 つ以上のレコードを前記新しいレコードに置き換えて、前記構造化文書を変換する変換手段と、

変換後の構造化文書を出力する文書出力手段と
を備えることを特徴とする変換装置。

【請求項 2】 検索キーを入力するキー入力手段と、前記変換後の構造化文書を該検索キーで検索し、あるレコードの要素の内容から前記検索キーに対応する文字列が検出されたとき、該あるレコードにおける他の要素の内容から、検出された文字列の位置に対応する文字列を抽出し、該検出された文字列と抽出された文字列から該検索キーを含む変換前のレコードを復元し、検索結果として出力する検索手段とをさらに備えることを特徴とする請求項 1 記載の変換装置。

【請求項 3】 階層構造の要素の集合で記述された構造化文書の情報を入力する文書入力手段と、

前記構造化文書の情報を格納する格納手段と、

前記構造化文書において、ある要素の 1 段下の層で連続して並ぶ同じ要素名の要素同士の組み合わせと、該組み合わせの各要素より下位のある層の同じ要素名の要素同士であって、該組み合わせの各要素から該ある層に至る経路上の各層において互いに同じ要素名の要素を経由するような、該ある層の要素同士の組み合わせとに含まれる各要素の内容を合成対象として接合し、複数の新しい要素を生成する接合手段と、

前記複数の新しい要素を含み、該複数の新しい要素の間に元の要素の相対的位

置関係を継承した合成部分構造を生成する生成手段と、

接合されなかった要素より上位の要素から生成された合成部分構造に含まれる新しい要素の下位に、該接合されなかった要素の複製を生成する複製手段と、

不要な元の要素を削除する削除手段と、

前記接合手段、生成手段、複製手段、および削除手段を用いて、前記構造化文書を合成部分構造から成る合成型構造化文書に変換する変換手段と、

前記合成型構造化文書を出力する文書出力手段とを備えることを特徴とする変換装置。

【請求項 4】 前記生成手段は、前記ある層に至る経路上の 2 つ以上の層において、連続して並ぶ同じ要素名の要素同士の組み合わせが見られないとき、前記合成部分構造を生成することを特徴とする請求項 3 記載の変換装置。

【請求項 5】 コンピュータのためのプログラムを記録した記録媒体であって、該プログラムは、

階層構造の要素の集合で記述され、それぞれが 1 つ以上の要素を含む複数のレコードから成る構造化文書の 2 つ以上のレコード間で、相対的に同じ位置にある要素の内容を接合して、新しい要素を生成し、

前記新しい要素を含み、前記 2 つ以上のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成し、

前記 2 つ以上のレコードを前記新しいレコードに置き換えて、前記構造化文書を変換する

処理を前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、階層構造を持つ要素の集合で記述される構造化文書の検索処理に係り、構造化文書の要素を検索するためにその文書の構造を変換する変換装置に関する。

【0002】

【従来の技術】

構造化文書の記述形式の代表例としては、大規模データベース向けの SGML (Standard General Markup Language)、WWW (World Wide Web) 向けに簡便な構成を持つ HTML (Hyper Text Markup Language)、SGML をインターネット向けに簡略化した XML (eXtensible Markup Language) 等がある。HTML は、WWW のコンテンツ形式として世界的に普及している。XML は、HTML を補うものとして、最近、特に注目を浴びており、インターネット上で文書を記述するだけでなく、携帯電話、カーナビゲーション等あらゆる情報機器が交信するための媒介となりつつある。

【0003】

XML で記述された XML 文書の概要については、例えば、「標準 XML 完全解説」(技術評論社、1998 年、pp. 27-51) に紹介されている。XML 文書は、図 20 に示すように、大きく分けて、XML 宣言 11、文書型定義 (Document Type Definition, DTD) 12、および XML 実現値 (インスタンス) 13 の 3 つの部分から成る。このうち、XML 実現値の部分は、階層構造を持つ要素の集合で記述され、これらの要素を識別するマークとして、タグが用いられる。

【0004】

図 21 は、1 つの要素を表すタグの書き方を示している。図 21 において、要素名を含む開始タグ 21 と終了タグ 22 の間に記述された“要素の内容です。”の部分が要素の内容を表し、空要素タグ 23 は、内容のない要素のタグを表す。また、要素の内容として平文と下位の要素が混在するような階層構造を表すタグの書き方は、図 22 のようになる。図 22 においては、要素 a の内容 1 と内容 2 の間に要素 b が挿入されており、要素 a の下位に要素 b が存在する。この場合、要素 a と要素 b は親子関係にある。

【0005】

さらに、要素に属性が与えられている場合は、以下に示すように、その要素の開始タグに属性名と属性値が記述される。

<要素名 属性名1 = “属性値1” 属性名2 = “属性値2” . . . >

また、XML文書は、処理上の観点から、整形式（well-formed）と検証済み（valid）の2つの種類に分けられる。このような2種類のXML文書を含む構造化文書の構成と処理上の区分との関係は、図23のようになる。図23では、整形式XML文書、検証済みXML文書、SGML文書、およびHTML文書のそれぞれについて、宣言、文書型定義、および実現値が必須である（○）か否（△）かが示されている。例えば、整形式XML文書の場合は、実現値のみが必須であり、宣言と文書型定義はなくてもよい。

【0006】

XML文書を解析して、ブラウザ等の他の応用ソフトウェアに渡す媒介となる役割を果たすソフトウェアは、XMLプロセッサ（XMLパーサー）と呼ばれる。XMLプロセッサの概要については、例えば、「OpenDesign2月号」（CQ出版、2000年2月、pp. 39-85）に紹介されている。

【0007】

図24は、XMLプロセッサが行う処理の例を示している。図24において、XMLプロセッサ32は、与えられたXML文書31をチェックして、木構造で表されたXML文書33を応用ソフトウェア34に渡す。このとき、XML文書31に文書型定義が含まれていなければ、XML実現値のタグ付け形式のみがチェックされる。

【0008】

このようなXMLプロセッサにおいて、Java（商標）言語でXML文書进行操作するためのAPI（Application Programming Interface）には、SAX（Simple API for XML）とDOM（Document Object Model）の2種類がある。SAXは、XML文書を読みながら、文書や要素の開始や終了、文字列の出現といった事象（event）を応用ソフトウェアに通知する事象駆動型のAPIである。

【0009】

これに対して、DOMは、汎用的なXML操作APIであり、XML文書をDOMオブジェクトの木構造として、メモリ上に展開する。そして、応用ソフトウ

エアは、このDOMオブジェクトに対する操作を行うことで、XML文書へのアクセスを行うことができる。また、DOMオブジェクトから元のXML文書を生成することもできる。

【0010】

例えば、図25のようなXML文書からは、図26のようなDOMの木構造が生成される。図26において、矢印は、各ノードを呼び出すためのメソッド（関数）を表し、Document 41は、XML文書の全体を表現するインタフェースに対応する。

【0011】

また、NodeList 42は、あるノードに属する下位の要素や文字データをXML文書内での出現順に管理するために使用され、Element 43やText 44等のインスタンスを下位のノードとして持つ。NamedNodeMap 45は、並び順に意味はないが、名前をキーにして値を参照する必要があるようなノードを収容するためのコレクションであり、ここには属性（Attr 46）等が記述される。

【0012】

XML文書の代表的な応用例として、データベースとしてのXML文書のタグ検索がある。この処理では、XML文書で与えられた検索キーに対応する箇所が検索され、検索結果が出力される。

【0013】

図27は、DOMを用いた場合のタグ検索処理のフローチャートである。処理プログラムは、まず、XML文書を入力し（ステップS1）、検索キーを入力する（ステップS2）。次に、XMLプロセッサのインスタンスを生成し（ステップS3）、それを実行する（ステップS4）。これにより、XML文書のタグ構造が解析され、DOMの木構造が構築される。

【0014】

次に、木構造をルート（根）から辿り、検索キーに対応する個所を検出して、木構造の不要な部分を削除する（ステップS5）。これにより、木構造のノードが削減されて、部分木が生成される。そして、得られた部分木を検索結果として

出力し（ステップ S 6）、処理を終了する。

【0015】

大規模データベースをXMLで構築した場合、図27のタグ検索は、比較的高速に検索できる点で有効な方法である。例えば、住民票データのデータベースにおいて、出身県を検索キーとして入力し、DOMの木構造を探索して、該当する個人データの部分木を残して出力するような処理が可能になる。

【0016】

また、図28は、SAXを用いた場合のタグ検索処理のフローチャートである。処理プログラムは、まず、XML文書を入力し（ステップS11）、検索キーを入力する（ステップS12）。次に、ハンドラーのインスタンスを生成し（ステップS13）、XMLプロセッサのインスタンスを生成して（ステップS14）、XMLプロセッサを実行する（ステップS15）。

【0017】

XMLプロセッサは、XML文書のタグ構造を解析し、タグを検出する度にハンドラーを実行して、検索キーに対応する個所を検出する（ステップS16）。そして、得られた検索結果を出力して（ステップS17）、処理を終了する。

【0018】

【発明が解決しようとする課題】

しかしながら、上述した従来のDOMを用いたタグ検索には、以下のような問題がある。

【0019】

DOMの木構造の規模が大きくなると、木構造を辿って、各要素の内容から検索キーと同じ文字列を検出するために、多大な処理時間を要する。また、DOMでは、各項目に長い文字列が出現することを予期して、長い固定長メモリ領域を確保してデータを書き込むので、木構造が大きくなると、大きな動作メモリ量が必要となる。

【0020】

本発明の課題は、XML文書のような構造化文書を変換することにより、タグ検索の処理速度を向上させ、必要な動作メモリ量を削減する変換装置を提供する

ことである。

【 0 0 2 1 】

【課題を解決するための手段】

図 1 は、本発明の変換装置の原理図である。本発明の第 1 の局面において、変換装置は、文書入力手段 5 1、接合手段 5 2、生成手段 5 3、変換手段 5 4、および文書出力手段 5 5 を備える。

【 0 0 2 2 】

文書入力手段 5 1 は、階層構造の要素の集合で記述され、それぞれが 1 つ以上の要素を含む複数のレコードから成る構造化文書の情報を入力する。接合手段 5 2 は、構造化文書の 2 つ以上のレコード間で、相対的に同じ位置にある要素の内容を接合して、新しい要素を生成する。

【 0 0 2 3 】

生成手段 5 3 は、生成された新しい要素を含み、上記 2 つ以上のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成する。変換手段 5 4 は、それらの 2 つ以上のレコードを新しいレコードに置き換えて、構造化文書を変換する。そして、文書出力手段 5 5 は、変換後の構造化文書を出力する。

【 0 0 2 4 】

レコードは、構造化文書を構成するために繰り返される単位データに対応し、構造化文書は、複数のレコードから成る。文書入力手段 5 1 により入力された構造化文書は、接合手段 5 2 に渡される。そして、接合手段 5 2 は、2 つ以上のレコードの同じ位置にある要素を合成対象としてそれらの内容を接合し、接合された内容を持つ新しい要素を生成して、生成手段 5 3 に渡す。

【 0 0 2 5 】

次に、生成手段 5 3 は、受け取った新しい要素を用いて、接合前の元のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成し、変換手段 5 4 に渡す。変換手段 5 4 は、元のレコードを新しいレコードに置き換えて、変換後の構造化文書を生成し、文書出力手段 5 5 に渡す。そして、文書出力手段 5 5 は、受け取った構造化文書を変換結果として出力する。

【 0 0 2 6 】

また、本発明の第 2 の局面において、変換装置は、文書入力手段 5 1、接合手段 5 2、生成手段 5 3、変換手段 5 4、文書出力手段 5 5、格納手段 5 6、複製手段 5 7、および削除手段 5 8 を備える。

【 0 0 2 7 】

文書入力手段 5 1 は、階層構造の要素の集合で記述された構造化文書の情報を入力し、格納手段 5 6 は、構造化文書の情報を格納する。接合手段 5 2 は、構造化文書において、ある要素の 1 段下の層で連続して並ぶ同じ要素名の要素同士の組み合わせと、その組み合わせの各要素より下位のある層の同じ要素名の要素同士であって、その組み合わせの各要素からある層に至る経路上の各層において互いに同じ要素名の要素を経由するような、ある層の要素同士の組み合わせとに含まれる各要素の内容を合成対象として接合し、複数の新しい要素を生成する。

【 0 0 2 8 】

生成手段 5 3 は、生成された複数の新しい要素を含み、それらの新しい要素の間で元の要素の相対的位置関係を継承した合成部分構造を生成する。複製手段 5 7 は、接合されなかった要素より上位の要素から生成された合成部分構造に含まれる新しい要素の下位に、接合されなかった要素の複製を生成する。削除手段 5 8 は、不要な元の要素を削除する。

【 0 0 2 9 】

変換手段 5 4 は、接合手段 5 2、生成手段 5 3、複製手段 5 7、および削除手段 5 8 を用いて、構造化文書を合成部分構造から成る合成型構造化文書に変換する。そして、文書出力手段 5 5 は、合成型構造化文書を出力する。

【 0 0 3 0 】

文書入力手段 5 1 により入力された構造化文書は、格納手段 5 6 に格納される。次に、接合手段 5 2 は、格納手段 5 6 から構造化文書の情報を取り出し、合成対象の要素を選択する。ここでは、ある要素の 1 段下の層に並んでいる兄弟要素のうち、連続して並ぶ複数の同名の要素の組み合わせが第 1 の合成対象として選択される。また、それらの要素より下位の任意の層に複数の互いに同名の要素が存在し、その層に至る経路上においても互いに同名の要素が連なっているとき、その任意の層の同名の要素の組み合わせが第 2 の合成対象として選択される。

【 0 0 3 1 】

次に、接合手段 5 2 は、各合成対象の組み合わせの中で同名の要素の内容を接合し、接合された内容を持つ新しい要素を生成して、生成手段 5 3 に渡す。生成手段 5 3 は、受け取った新しい要素を含み、それらの要素の間で元の要素の相対的位置関係を継承した合成部分構造を生成して、複製手段 5 7 に渡す。

【 0 0 3 2 】

複製手段 5 7 は、接合されなかった要素が存在する場合、その要素より上位の要素から生成された合成部分構造に含まれる、新しい要素の下位に、その要素の複製を追加する。また、削除手段 5 8 は、接合された元の要素と複製された元の要素を削除する。変換手段 5 4 は、接合手段 5 2、生成手段 5 3、複製手段 5 7、および削除手段 5 8 を制御することにより、元の構造化文書を合成型構造化文書に変換し、文書出力手段 5 5 に渡す。そして、文書出力手段 5 5 は、受け取った合成型構造化文書を変換結果として出力する。

【 0 0 3 3 】

このような変換装置によれば、構造化文書の複数の要素が合成されて 1 つになるため、文書情報が圧縮され、文書を格納するためのメモリ量が削減される。また、要素の数が減ることによって木構造のノードが減少するので、タグ検索の処理速度が向上する。

【 0 0 3 4 】

また、変換後の構造化文書においても、元の要素の相対的位置関係が継承されるため、元の階層構造を把握することができ、既存のブラウザ、ビューア等の応用ソフトウェアを適用して、従来の機能をそのまま実行することができる。言い換えれば、既存の応用ソフトウェアから見て、元の文書を変換したことが分からないような透過性が実現される。

【 0 0 3 5 】

例えば、図 1 の文書入力手段 5 1 は、後述する図 1 8 の入力装置 8 3 またはネットワーク接続装置 8 7 に対応し、図 1 の格納手段 5 6 は、図 1 8 のメモリ 8 2 または外部記憶装置 8 5 に対応する。また、例えば、図 1 8 の接合手段 5 2、生成手段 5 3、変換手段 5 4、文書出力手段 5 5、複製手段 5 7、および削除手段

58は、図18のCPU（中央処理装置）81およびメモリ82に対応する。

【0036】

【発明の実施の形態】

以下、図面を参照しながら、本発明の実施の形態を詳細に説明する。

まず、図2のXML文書进行处理対象として、合成の対象を指定する処理について説明する。一般に、XML文書は、複数のレコードの繰り返しにより構成される。例えば、図2のXML文書では、`<event>`から次の`</event>`までの部分が1つのレコードに対応する。図2のXML文書をXMLプロセッサで解析すると、図3のようなDOMの木構造が得られる。ここでは、図2の8個のEvent要素のうち、最初の4個のみが明示的に示されている。

【0037】

木構造において、一般に、合成の対象となるのは、同じ名前を持つ兄弟の要素からそれぞれ派生した部分木である。図3の木構造を見ると、Eventlist要素61の下には、Event要素62が4個連続して存在する。各Event要素62に連なる部分木は、1つのレコードに対応する。このうち、左から2番目のEvent要素62の下には、Info要素63が2個連続して存在し、1番目、3番目、および4番目のEvent要素62では、Info要素63はそれぞれ1個しか存在しない。

【0038】

2番目のEvent要素62に連なる部分木の中では、子であるInfo要素63が2つ存在し、それぞれのInfo要素63が部分木を成している。このようなEvent要素62を合成の対象にすると、合成の候補となる部分木が重なりあい、処理が煩雑になる。

【0039】

そこで、このEvent要素62の部分木のように、2つ以上の階層にまたがって同名の兄弟要素が連続して並ぶような場合は、合成の対象から外すことにする。したがって、連続して並ぶ同名の兄弟要素のうちの1つからある層に至る経路上の2つ以上の層において、連続して並ぶ同名の兄弟要素の組み合わせが見られない場合に、それらの兄弟要素に連なる部分木が合成の対象として指定される

【 0 0 4 0 】

実際の処理では、変換装置は、各要素の `NodeList` を見て、兄弟のノードの中で、同じ要素名を持つ要素が連続して出現するようなものを検出し、合成の対象から外す。図 3 では、`Event` 要素 6 2 が 4 つ連続しているが、2 番目の `Event` 要素 6 2 は、下の層に 2 つの連続する `Info` 要素 6 3 を持っているので、対象外になる。すると、連続した `Event` 要素 6 2 は 3 番目および 4 番目のみとなり、これらにそれぞれ連なる部分木が合成の対象として指定される。

【 0 0 4 1 】

このように、合成の対象となる部分木が複数個得られた場合、これらの部分木は幾つかの群（グループ）に分割される。2 個の部分木を 1 つの群にまとめた場合、図 3 では、1 番目および 2 番目の `Event` 要素 6 2 の部分木から成る部分 `P 1` が合成の対象外となり、3 番目および 4 番目の `Event` 要素 6 2 の部分木から成る部分 `P 2` が合成の対象となる。同様にして、図 2 の他の `Event` 要素のうち、`P 3` および `P 4` の部分がそれぞれ合成の対象となる。

【 0 0 4 2 】

また、各部分木の親の要素（ここでは、`Event` 要素）の間には、親の要素の内容である文字データ（`Text`）が挿入されている。これらの文字データについても、合成の対象となった `Event` 要素 6 2 に付随するものを 2 つずつ群に分けて、接合の対象とする。

【 0 0 4 3 】

次に、図 4 の XML 文書を処理対象として、合成対象として指定された要素を合成する処理について説明する。ここでは、部分木を 4 つずつ群にまとめて、各群の中の要素を合成するものとする。図 4 の XML 文書を XML プロセッサで解析すると、図 5 のような DOM の木構造が得られる。

【 0 0 4 4 】

図 5 の木構造において、変換装置は、互いの先祖が同じ要素名を持ち、それ自身も同じ要素名を持つようなノード同士で、要素内容を接合して登録する。ここ

では、3つのStart要素73が同じ要素名のEvent要素72を親として持っているので、それらの要素内容である“8:40”、“9:00”、および“9:30”が接合される。

【0045】

同様に、4つのInfo要素74がEvent要素72を親として持っているので、それらの要素内容である“出社”、“退社”、“接客”、および“会議”も接合される。このとき、各Event要素72に付随するText（親であるEventlist要素71の内容）も接合されて登録される。

【0046】

接合される2つの要素内容の間には、境界を示すために境界文字（デリミッタ）を挿入する。一方の部分木には存在し、他方の部分木には存在しない（つまり、欠損している）要素については、接合の際に、存在している要素内容の後に境界文字だけを追加して、要素の欠損を表すことにする。この場合、新しい要素の内容には、境界文字が連続して挿入されることになる。

【0047】

こうして、図5の部分木を合成した後の木構造は、図6のようになる。図6において、Event要素72に連なる部分木が合成部分木に対応し、Text75が合成部分木の要素内容を表し、“@”が境界文字に対応する。図5では、左から2番目のEvent要素72にStart要素73が欠損しているので、図6のStart要素73の要素内容“8:40@@9:00@9:30”において、2つの@が連続して挿入されている。

【0048】

また、図6の木構造に対応する合成後のXML文書は、図7のようになる。図7のXML文書では、<event>から次の</event>までの部分が合成部分木のレコードに対応する。このレコードは、要素を合成して得られた合成部分構造を表す。

【0049】

図6および図7に示されるように、合成部分木においては、要素内容が出現順に接合され、要素内容の間に境界文字@が挿入される。また、接合すべき内容が

ない場合は、空白を入れずに次の@が挿入される。このような接合方法によれば、境界文字@の位置と個数から元の部分木の構造を復元することができる。例えば、Text 75において“A@B”と記述されていれば、初めの2つのEvent要素72に対応する要素内容が“A”および“B”であることが分かる。

【0050】

図6では、Event要素72の内容“第一@第二@第三@第四”は、1番目、2番目、3番目、および4番目のEvent要素72の内容が、それぞれ、“第一”、“第二”、“第三”、および“第四”であることを表す。また、Info要素74の内容“出社@退社@接客@会議”は、1番目、2番目、3番目、および4番目のEvent要素72に連なるInfo要素74の内容が、それぞれ、“出社”、“退社”、“接客”、および“会議”であることを表す。

【0051】

一方、Start要素73の内容“8:40@@9:00@9:30”は、1番目、3番目、および4番目のEvent要素72に連なるStart要素73の内容が、それぞれ、“8:40”、“9:00”、および“9:30”であり、2番目のEvent要素72にはStart要素73が欠損していることを表す。したがって、このような合成部分木の要素内容から、図5の木構造を容易に復元することができる。

【0052】

また、合成部分木のデータを保存する場合、変換装置は、その部分木の親から任意の要素に至るまでに経由する要素名の組み合わせと、接合した要素内容の文字列を、テーブルに登録する。このような保存方法によれば、異なる部分木に属する要素でも、経由する要素名の組み合わせが等しければ、テーブルの同じ欄にそれらの要素内容を登録することができ、接合が可能になる。

【0053】

図8は、図6の合成部分木に登録したハッシュ表の例を示している。図8のハッシュ表では、ハッシュ値H1の欄に、Info要素の接合された要素内容が、親の要素名Eventと要素名Infoの組み合わせとともに登録されている。また、ハッシュ値H2の欄には、Start要素の接合された要素内容が、要素

名 Event と Start の組み合わせとともに登録されている。

【 0 0 5 4 】

例えば、ハッシュ値 H 1 および H 2 は、それぞれ、対応する要素名の組み合わせに基づいて算出される。また、こうして生成されたハッシュ表を元にして、DOM の木構造に合成部分木が追加される。

【 0 0 5 5 】

上述した合成処理では、XML 文書の構造が一般的な木構造である場合を想定しているが、完全に同じ構造の部分木の繰返しにより木構造が構成されている場合は、より簡便な処理を採用することができる。この場合、XML 文書を DOM の木構造に変換して要素同士の位置関係を解析しなくても、あらかじめ要素の位置関係をソフトウェアに記述しておけば、内容を接合することができる。そこで、変換装置は、記述された位置関係からハッシュ値を計算して、ハッシュ表の該当個所に要素内容を登録する。

【 0 0 5 6 】

図 9 は、このような簡易型合成処理で用いられるハッシュ表の例を示している。簡易型合成処理は、同じ部分構造が規則的に繰り返される文書に対して適用されるため、要素の欠損がないことを前提としている。図 9 のハッシュ表では、ハッシュ値 H 3 の欄に、図 8 の Info 要素と同様の要素内容が登録されているが、ハッシュ値 H 4 の欄には、図 8 の Start 要素とは異なり、欠損のない要素内容が登録されている。例えば、ハッシュ値 H 3 および H 4 は、それぞれ、対応する要素の位置または要素名等に基づいて算出される。

【 0 0 5 7 】

次に、合成後の XML 文書を用いたタグ検索の例について説明する。合成後の XML 文書のタグ検索において、変換装置は、文書内の要素の内容に含まれる 2 つの境界文字の間の文字列と検索キーの文字列とを照合し、検索キーに対応する文字列を検出する。次に、検出された文字列の前にある境界文字の順位を求め、同じ合成部分木における他の要素の内容において、その順位に対応する境界文字と次の境界文字の間の文字列を抽出する。そして、これらの文字列から合成前の XML 文書の対応する部分を復元し、検索結果として出力する。

【0058】

図10に示すXML文書を処理対象とした場合、変換装置は、同じ要素名“個人”の要素を上位に持ち、かつ、それ自身が同じ要素名を持つような要素に属する内容同士を、境界文字を用いて接合する。これにより、“姓”、“名”、および“旧姓”の要素の内容が接合される。そして、元の要素のノードを消去すると、図11のようなXML文書が生成される。

【0059】

次に、生成されたXML文書のタグ検索において、ユーザが“鈴木”を検索キーとして指定すると、変換装置は、要素内容に“鈴木”を含む群の合成部分木を検索する。図11の文書では、“鈴木”を含む群が1つだけしか存在しないので、この文書の全体が検索結果として得られる。次に、変換装置は、得られた結果において、“鈴木”に対応する部分だけを残して、他の部分は削除する。その結果、“佐藤”に対応する部分が削除され、図12のような検索結果が出力される。

【0060】

このような変換処理によれば、XML文書の複数の要素が合成されて1つになるため、文書情報が大幅に圧縮される。また、要素の数が減ることで木構造のノードが減少するので、タグ検索に要する探索時間が大幅に削減される。

【0061】

次に、図13から図17までを参照しながら、変換装置が行う処理についてより詳細に説明する。

図13は、XML文書の変換を含むタグ検索処理のフローチャートである。まず、ユーザは、1つの群にまとめられるレコードの数を指定する数値nを変換装置に入力する（ステップS21）。この数値nは、要素の合成処理において、レコードを組み合わせて群にまとめる単位として用いられる。次に、ユーザは、処理対象となるXML文書を入力する（ステップS22）。

【0062】

次に、変換装置は、合成対象指定処理を行って、入力されたXML文書の中で、合成するn個のレコードの組み合わせ（群）を指定し（ステップS23）、要

素の合成処理を行う（ステップ S 2 4 ～ S 2 7）。

【 0 0 6 3 】

合成処理において、変換装置は、まず、指定されたすべての組み合わせについての合成が終了したか否かをチェックする（ステップ S 2 4）。合成されていない組み合わせがあれば、その組み合わせに含まれる n 個のレコードの間で、相対的に同じ位置関係にある要素の内容を接合し、1 つ以上の新しい要素を生成する（ステップ S 2 5）。

【 0 0 6 4 】

次に、生成された新しい要素を元に、元のレコードと同じような要素の相対的位置関係を継承した新しいレコードを生成する（ステップ S 2 6）。そして、元の n 個のレコードを削除して、新しいレコードに置き換え（ステップ S 2 7）、ステップ S 2 4 以降の処理を繰り返す。

【 0 0 6 5 】

ステップ S 2 4 において、すべての組み合わせについての合成が終了すると、復元処理を行って（ステップ S 2 8）、処理を終了する。この復元処理では、合成処理により変換された文書を検索キーで検索し、あるレコードで検索キーと同じ部分文字列を内容に含む要素が検出されると、そのレコードにおける他の各要素の内容においても、検出された部分文字列の位置に対応する部分文字列を抽出する。そして、これらの部分文字列から、検索キーを含む変換前の複数個のレコードを復元し、検索結果として出力する。

【 0 0 6 6 】

次に、図 1 4 は、図 1 3 のステップ S 2 3 で行われる合成対象指定処理のフローチャートである。この処理では、兄弟要素の中で同じ名前の要素が複数個連続する場合、それらの要素を所定数の要素から成る複数の群に分割し、各群に含まれる要素に基づいて、合成対象が指定される。

【 0 0 6 7 】

変換装置は、まず、同名の兄弟要素を n 個単位の群に分けるために、数値 n を合成対象指定処理に入力する（ステップ S 3 1）。次に、XML 文書に XML プロセッサを適用して、木構造から成る XML 文書のオブジェクト（DOM）を算

出し（ステップ S 3 2）、要素数の集計処理を行う（ステップ S 3 3～S 3 5）。

【 0 0 6 8 】

この処理では、まず、木構造のすべての要素に関する集計が終了したか否かをチェックする（ステップ S 3 3）。集計が終了していなければ、木構造から各要素に属する Node List（連なるノードの一覧）を取得してメモリに格納する（ステップ S 3 4）。そして、各 Node List 内で兄弟の関係にある要素のノードにおいて、同じ要素名のものが連続して出現する回数を集計し（ステップ S 3 5）、ステップ S 3 3 以降の処理を繰り返す。例えば、要素 a、要素 a、要素 b が順に出現した場合、要素 a の連続出現回数は 2 回となる。

【 0 0 6 9 】

ステップ S 3 3 において、すべての要素に関する集計が終了すると、次に、木構造の兄弟の中で連続する同じ要素名の要素の各々について、その祖先または子孫のノードにおいて、やはり、兄弟の関係にある同じ要素名の連続があるか否かをチェックする（ステップ S 3 6）。

【 0 0 7 0 】

そのような要素名の連続が検出されなければ、現在注目している層の兄弟の中でのみ同じ要素名の連続が存在することになるので、連続するそれらの要素を合成対象として n 個ずつの群に分割し、n 個の各ノードの位置を記録する（ステップ S 3 7）。そして、得られた n 個ずつの要素の群とそれらの位置情報を出力し（ステップ S 3 8）、処理を終了する。

【 0 0 7 1 】

また、ステップ S 3 6 において、兄弟の関係にある同じ要素名の連続が祖先または子孫から検出されれば、注目している層を含む複数の層において同じ要素名の兄弟要素の連続が存在することになるので、そのような祖先または子孫を持つ要素を合成対象から外す。

【 0 0 7 2 】

次に、図 1 5 は、図 1 3 のステップ S 2 4～S 2 7 で行われる合成処理のフローチャートである。変換装置は、まず、合成対象指定処理から出力された、n 個

ずつの同名の兄弟要素の群とそれらの位置情報を、合成処理に入力し（ステップ S 4 1）、入力されたすべての群についての合成が終了したか否かをチェックする（ステップ S 4 2）。

【 0 0 7 3 】

合成が終了していない群があれば、次に、その群のすべての同名の兄弟要素 a についての処理が終了したか否かをチェックする（ステップ S 4 3）。そして、処理が終了していない要素 a があれば、その要素 a に連なる部分木を部分木 a として、部分木 a の要素を探索する（ステップ S 4 4 ～ S 5 1）。同じ群に属する要素名 a の 2 つの要素を仮に a 1、a 2 と呼ぶことにすると、それらの要素に連なる部分木は、それぞれ部分木 a 1、部分木 a 2 となる。

【 0 0 7 4 】

部分木 a の探索処理では、まず、部分木 a のルートである要素 a から下のすべての要素を探索したか否かをチェックする（ステップ S 4 4）。探索していない任意の要素 c があれば、要素 a からその要素 c に至るまでの経路を探索し、経路上の要素の名前の文字列を要素の流れとして記録する（ステップ S 4 5）。

【 0 0 7 5 】

例えば、要素 a、b、c の順に上から辿った場合、（要素 a）+（要素 b）+（要素 c）が要素の流れとして記録される。部分木 a 1 に属する要素の流れ（要素 a）+（要素 b）+（要素 c）と、部分木 a 2 に属する要素の流れ（要素 a）+（要素 b）+（要素 c）は、同じ要素の流れになる。

【 0 0 7 6 】

次に、記録された要素の流れの文字列を元に、ハッシュ関数の値（要素 c のハッシュ値）を計算する（ステップ S 4 6）。ここでは、例えば、要素名 a、b、c の文字列を文字符号に変換して得られる整数を、それぞれ I a、I b、I c として、 $I a * 256$ 、 $I b * 16$ 、および I c の排他的論理和 $EX(a b c)$ を求め、次式によりハッシュ値を計算する。

$$\text{ハッシュ値} = EX(a b c) \% m$$

ただし、 m は、256に対して素な整数であり、 $\%m$ は、 m による剰余演算を表す。このとき、部分木 a_1 と部分木 a_2 にそれぞれ属する同じ要素の流れ同士は、互いに同じハッシュ値を持つことになる。

【0077】

次に、現在の部分木における要素 c が、同じ群の中で処理済みの一連の部分木 a にはなかった新しい要素か否かをチェックする（ステップ S47）。要素 c が新しい要素であれば、ハッシュ表における要素 c のハッシュ値の欄に、それまでに処理した部分木 a の数（要素 a の数）だけ、境界文字 $@$ を並べて登録する（ステップ S48）。 $@$ の数が多い場合は、連長表現を用いてもよい。

【0078】

次に、同じ群の中で処理済みの一連の部分木 a とは異なり、現在の部分木において要素 c が欠損しているか否かをチェックする（ステップ S49）。要素 c が欠損していれば、ハッシュ表における要素 c のハッシュ値の欄に、境界文字 $@$ を追加して登録する（ステップ S50）。また、要素 c が欠損していなければ、ハッシュ表における要素 c のハッシュ値の欄に、要素 c の内容と境界文字 $@$ を追加して登録する（ステップ S51）。そして、ステップ S44 以降の処理を繰り返す。

【0079】

ステップ S46～S52 の処理により、異なる部分木に属する要素でも、要素の流れが等しいもの同士は、同じハッシュ値を持つことになり、それらの要素の内容は、同じハッシュ表の同じハッシュ値に対応する欄で、境界文字 $@$ を用いて接合される。

【0080】

ステップ S44 において、すべての要素の探索が終了すると、ステップ S43 以降の処理を繰り返す。そして、ステップ S43 において、すべての同名の兄弟要素 a についての処理が終了すると、同じ群に属する n 通りの部分木の接合が終了したことになる。そこで、次に、得られたハッシュ表の登録内容を元に、合成部分木を生成する（ステップ S52）。このとき、接合されなかった要素があれば、それより上位の要素から生成された合成部分木に含まれる、新しい要素の下

位に、その接合されなかった要素の複製を生成する。

【 0 0 8 1 】

次に、同じ群に属する処理済みの n 通りの部分木を削除し、合成部分木のみを残して（ステップ S 5 3）、ステップ S 4 2 以降の処理を繰り返す。そして、ステップ S 4 2 において、すべての群についての合成が終了すると、処理を終了する。このような合成処理によれば、複数の対応する要素内容が接合され、元の要素内容が削除されるので、文書が圧縮される。圧縮された文書は、外部記憶装置等に保存される。

【 0 0 8 2 】

次に、図 1 6 は、圧縮された XML 文書の上でタグ検索を行い、検索結果を出力する復元処理のフローチャートである。変換装置は、まず、圧縮された XML 文書を復元処理に入力し（ステップ S 6 1）、ユーザから指定された検索キーの文字列を復元処理に入力する（ステップ S 6 2）。

【 0 0 8 3 】

次に、図 2 7 と同様の処理により、圧縮された XML 文書の上でタグ検索を行う（ステップ S 6 3）。上述した合成処理では、元の XML 文書で同じ要素名 a を持つ連続した兄弟要素に連なる部分木を合成したが、基本的な要素の位置関係は変化していないため、透過性は保たれている。

【 0 0 8 4 】

合成部分木の要素内容の中で、検索キーと同じ部分文字列が検出された場合、その文字列の前に登録されている境界文字 $@$ の個数 d を計算する（ステップ S 6 4）。この d の値により、合成前の各部分木のうち、最初から $d + 1$ 番目の部分木に検索キーに相当する部分文字列があったことが分かる。

【 0 0 8 5 】

このとき、同じ要素内容の中の複数の個所で検索キーと同じ部分文字列が検出されれば、検出されたすべての文字列に対する d の値を計算する。 k 個の同じ部分文字列 ($0 < k \leq n$) が検出された場合、各々に対応する境界文字 $@$ の個数として d_1, \dots, d_k が計算される。

【 0 0 8 6 】

次に、同名の複数の要素 a に連なる合成部分木のうち、検索キーと同じ部分文字列が検出された要素内容を含む合成部分木を、メモリ上に格納する（ステップ S 6 5）。以下の分離処理は、メモリ上に格納された合成部分木のみに対して行われるが、外部記憶装置に保存されている合成部分木は変更されない。

【0087】

次に、メモリ上のすべての合成部分木について処理を行ったか否かをチェックする（ステップ S 6 6）。未処理の合成部分木があれば、次に、未処理のある合成部分木のすべての要素を処理したか否かをチェックする（ステップ S 6 7）。

【0088】

未処理の要素があれば、その要素の内容において、 d_i 番目（ $i = 1, \dots, k$ ）の“@”と $d_i + 1$ 番目の“@”に囲まれた部分文字列を残して、残りの文字列を削除する（ステップ S 6 8）。そして、ステップ S 6 7以降の処理を繰り返す。このような文字列の削除処理は、木構造全体の一部分のみに対して行われるので、削除処理に伴う処理速度の劣化は防止される。

【0089】

これにより、すべての要素内容において、同じ d_i の値の組み合わせに対応する個所の部分文字列が抽出され、1つの合成部分木を構成する n 個の元の部分木のうち、検索キーを要素内容に含むものだけが再現される。

【0090】

ステップ S 6 7において、すべての要素の処理が終了すると、次に、再現された元の部分木をXML文書に変換し（ステップ S 6 9）、生成されたXML文書を出力して（ステップ S 7 0）、ステップ S 6 6以降の処理を繰り返す。そして、ステップ S 6 6において、メモリ上のすべての合成部分木の処理が終われば、復元処理を終了する。

【0091】

ところで、欠損等がなく、完全に同じ部分構造が繰り返される文書の場合、DOMを生成する必要はなく、図14の合成対象指定処理と図15の合成処理の代わりに、上述した簡易型合成処理を用いることができる。

【0092】

図17は、このような簡易型合成処理のフローチャートである。変換装置は、まず、数値 n を簡易型合成処理に入力し（ステップS71）、図4のEvent要素のように、同じ層で連続する要素名 a を入力する（ステップS72）。そして、要素名 a の出現頻度を表す変数 Fa を初期化して $Fa=0$ とおき（ステップS73）、登録に用いるハッシュ表の内容を初期化する（ステップS74）。

【0093】

次に、XML文書内のすべてのデータの走査が終了したか否かをチェックする（ステップS75）。走査されていない部分があれば、その部分から連続して存在する要素名 a を検出し（ステップS76）、 Fa に1を加算する（ステップS77）。

【0094】

次に、要素 a に連なる各要素 b を検出し（ステップS78）、ハッシュ表の要素 b のハッシュ値に対応する欄に、要素 b の内容と境界文字@を追加登録する（ステップS79）。ここでは、要素 b の親は要素 a であることが分かっているため、要素 b のハッシュ値の計算では、要素 a の文字列を省略することができる。そこで、例えば、要素 b の文字列を文字符号に変換して得られる整数を Ib として、次式によりハッシュ値が計算される。

$$\text{ハッシュ値} = Ib \% m$$

次に、 Fa が n の倍数になったか否かをチェックし（ステップS80）、 Fa が n の倍数でなければ、ステップS75以降の処理を繰り返す。これにより、複数の要素 a に連なる同じ層の要素の内容同士が接合される。

【0095】

ステップS80において、 Fa が n の倍数であれば、1つの群の合成処理が終了したことになるので、ハッシュ表に登録された、接合された要素内容を元に、部分的な文書構造を生成し、出力する（ステップS81）。次に、次の群の合成処理に備えて、ハッシュ表で要素 b のために使用された欄を初期化し（ステップS82）、ステップS75以降の処理を繰り返す。

【 0 0 9 6 】

そして、ステップ S 7 5 において、すべてのデータの走査が終了すると、処理を終了する。こうして生成された XML 文書の上でタグ検索を行う場合は、やはり、DOM を利用するのが有効であるので、上述した図 1 6 の復元処理が用いられる。

【 0 0 9 7 】

以上説明した実施形態においては、主として、XML 文書のタグ検索を例に用いているが、本発明は、XML 文書以外の構造化文書に対しても適用可能である。

【 0 0 9 8 】

本実施形態の変換装置は、例えば、図 1 8 に示すような情報処理装置（コンピュータ）を用いて構成される。図 1 8 の情報処理装置は、CPU（中央処理装置）8 1、メモリ 8 2、入力装置 8 3、出力装置 8 4、外部記憶装置 8 5、媒体駆動装置 8 6、およびネットワーク接続装置 8 7 を備え、それらはバス 8 8 により互いに接続されている。

【 0 0 9 9 】

メモリ 8 2 は、例えば、ROM (read only memory)、RAM (random access memory) 等を含み、処理に用いられるプログラムとデータを格納する。CPU 8 1 は、メモリ 8 2 を利用してプログラムを実行することにより、必要な処理を行う。

【 0 1 0 0 】

入力装置 8 3 は、例えば、キーボード、ポインティングデバイス、タッチパネル等であり、ユーザからの指示や情報の入力に用いられる。出力装置 8 4 は、例えば、ディスプレイ、プリンタ、スピーカ等であり、ユーザへの問い合わせや処理結果の出力に用いられる。

【 0 1 0 1 】

外部記憶装置 8 5 は、例えば、磁気ディスク装置、光ディスク装置、光磁気ディスク (magneto-optical disk) 装置、テープ装置等である。情報処理装置は、この外部記憶装置 8 5 に、上述のプログラムとデータを保存しておき、必要に応

じて、それらをメモリ 8 2 にロードして使用する。

【 0 1 0 2 】

媒体駆動装置 8 6 は、可搬記録媒体 8 9 を駆動し、その記録内容にアクセスする。可搬記録媒体 8 9 としては、メモリカード、フロッピーディスク、CD-ROM (compact disk read only memory)、光ディスク、光磁気ディスク等、任意のコンピュータ読み取り可能な記録媒体が用いられる。ユーザは、この可搬記録媒体 8 9 に上述のプログラムとデータを格納しておき、必要に応じて、それらをメモリ 8 2 にロードして使用する。

【 0 1 0 3 】

ネットワーク接続装置 8 7 は、LAN (local area network) 等の任意の通信ネットワークに接続され、通信に伴うデータ変換を行う。また、情報処理装置は、上述のプログラムとデータをネットワーク接続装置 8 7 を介して、サーバ等の他の装置から受け取り、必要に応じて、それらをメモリ 8 2 にロードして使用する。

【 0 1 0 4 】

図 1 9 は、図 1 8 の情報処理装置にプログラムとデータを供給することのできるコンピュータ読み取り可能な記録媒体を示している。可搬記録媒体 8 9 やサーバ 9 0 のデータベース 9 1 に保存されたプログラムとデータは、メモリ 8 2 にロードされる。そして、CPU 8 1 は、そのデータを用いてそのプログラムを実行し、必要な処理を行う。このとき、サーバ 9 0 は、プログラムとデータを伝送する伝搬信号を生成し、ネットワーク上の任意の伝送媒体を介して、情報処理装置に送信する。

(付記 1) 階層構造の要素の集合で記述され、それぞれが 1 つ以上の要素を含む複数のレコードから成る構造化文書の情報を入力する文書入力手段と、

前記構造化文書の 2 つ以上のレコード間で、相対的に同じ位置にある要素の内容を接合して、新しい要素を生成する接合手段と、

前記新しい要素を含み、前記 2 つ以上のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成する生成手段と、

前記 2 つ以上のレコードを前記新しいレコードに置き換えて、前記構造化文書

を変換する変換手段と、

変換後の構造化文書を出力する文書出力手段と
を備えることを特徴とする変換装置。

(付記 2) 検索キーを入力するキー入力手段と、前記変換後の構造化文書を該検索キーで検索し、あるレコードの要素の内容から前記検索キーに対応する文字列が検出されたとき、該あるレコードにおける他の要素の内容から、検出された文字列の位置に対応する文字列を抽出し、該検出された文字列と抽出された文字列から該検索キーを含む変換前のレコードを復元し、検索結果として出力する検索手段とをさらに備えることを特徴とする付記 1 記載の変換装置。

(付記 3) 階層構造の要素の集合で記述された構造化文書の情報を入力する文書入力手段と、

前記構造化文書の情報を格納する格納手段と、

前記構造化文書において、ある要素の 1 段下の層で連続して並ぶ同じ要素名の要素同士の組み合わせと、該組み合わせの各要素より下位のある層の同じ要素名の要素同士であって、該組み合わせの各要素から該ある層に至る経路上の各層において互いに同じ要素名の要素を経由するような、該ある層の要素同士の組み合わせとに含まれる各要素の内容を合成対象として接合し、複数の新しい要素を生成する接合手段と、

前記複数の新しい要素を含み、該複数の新しい要素の間で元の要素の相対的位置関係を継承した合成部分構造を生成する生成手段と、

接合されなかった要素より上位の要素から生成された合成部分構造に含まれる新しい要素の下位に、該接合されなかった要素の複製を生成する複製手段と、

不要な元の要素を削除する削除手段と、

前記接合手段、生成手段、複製手段、および削除手段を用いて、前記構造化文書を合成部分構造から成る合成型構造化文書に変換する変換手段と、

前記合成型構造化文書を出力する文書出力手段と
を備えることを特徴とする変換装置。

(付記 4) 前記生成手段は、前記ある層に至る経路上の 2 つ以上の層において、連続して並ぶ同じ要素名の要素同士の組み合わせが見られないとき、前記合成

部分構造を生成することを特徴とする付記 3 記載の変換装置。

(付記 5) 前記接合手段は、前記ある要素の 1 段下の層の要素同士の組み合わせを、所定数の要素から成る複数の群に分割し、各群に含まれる該所定数の要素に基づいて、前記合成対象を指定することを特徴とする付記 3 記載の変換装置。

(付記 6) 前記接合手段は、接合される 2 つの内容の間に境界文字を挿入して、前記新しい要素の内容を生成することを特徴とする付記 3 記載の変換装置。

(付記 7) 前記接合手段は、前記合成対象となる要素の内容が欠損しているとき、前記新しい要素の内容に前記境界文字を連続して挿入することを特徴とする付記 6 記載の変換装置。

(付記 8) 検索キーを入力するキー入力手段と、前記合成型構造化文書内の要素の内容に含まれる 2 つの境界文字の間の文字列と該検索キーの文字列とを照合し、ある合成部分構造の要素の内容から該検索キーに対応する文字列が検出されたとき、検出された文字列の前にある境界文字の順位を求め、該ある合成部分構造における他の要素の内容において、該順位に対応する境界文字と次の境界文字の間の文字列を抽出し、該検出された文字列と抽出された文字列から変換前の構造化文書の対応する部分を復元し、検索結果として出力する検索手段とをさらに備えることを特徴とする付記 6 記載の変換装置。

(付記 9) コンピュータのためのプログラムを記録した記録媒体であって、該プログラムは、

階層構造の要素の集合で記述され、それぞれが 1 つ以上の要素を含む複数のレコードから成る構造化文書の 2 つ以上のレコード間で、相対的に同じ位置にある要素の内容を接合して、新しい要素を生成し、

前記新しい要素を含み、前記 2 つ以上のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成し、

前記 2 つ以上のレコードを前記新しいレコードに置き換えて、前記構造化文書を変換する

処理を前記コンピュータに実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

(付記 10) コンピュータにプログラムを伝送する伝搬信号であって、該プロ

グラムは、

階層構造の要素の集合で記述され、それぞれが1つ以上の要素を含む複数のレコードから成る構造化文書の2つ以上のレコード間で、相対的に同じ位置にある要素の内容を接合して、新しい要素を生成し、

前記新しい要素を含み、前記2つ以上のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成し、

前記2つ以上のレコードを前記新しいレコードに置き換えて、前記構造化文書を変換する

処理を前記コンピュータに実行させることを特徴とする伝搬信号。

【0105】

【発明の効果】

本発明によれば、構造化文書の要素の数が減り、文書が圧縮されるため、構造化文書を格納するためのメモリ量が削減される。部分木の合成を行っても、要素間の基本的な関係は維持されるため、従来の応用ソフトウェアの処理に対する変換による影響はなく、処理の透過性が保たれる。

【0106】

また、木構造においてノードが減少するために、探索に要する処理時間が大幅に削減され、タグ検索の処理速度が向上する。例えば、オフラインで、元の構造化文書から合成部分木をあらかじめ生成しておけば、随時、その合成部分木をタグ検索に用いることができ、変換処理の時間はタグ検索の時間には含まれない。

【図面の簡単な説明】

【図1】

本発明の変換装置の原理図である。

【図2】

第1の処理対象のXML文書を示す図である。

【図3】

第1の処理対象の木構造を示す図である。

【図4】

第2の処理対象のXML文書を示す図である。

【図 5】

第 2 の処理対象の木構造を示す図である。

【図 6】

合成後の木構造を示す図である。

【図 7】

第 1 の合成後の XML 文書を示す図である。

【図 8】

第 1 のハッシュ表を示す図である。

【図 9】

第 2 のハッシュ表を示す図である。

【図 1 0】

第 3 の処理対象の XML 文書を示す図である。

【図 1 1】

第 2 の合成後の XML 文書を示す図である。

【図 1 2】

検索結果を示す図である。

【図 1 3】

XML 文書の変換を含むタグ検索処理のフローチャートである。

【図 1 4】

合成対象指定処理のフローチャートである。

【図 1 5】

要素合成処理のフローチャートである。

【図 1 6】

復元処理のフローチャートである。

【図 1 7】

簡易型合成処理のフローチャートである。

【図 1 8】

情報処理装置の構成図である。

【図 1 9】

記録媒体を示す図である。

【図 2 0】

XML 文書の構成部分を示す図である。

【図 2 1】

タグの書き方を示す図である。

【図 2 2】

要素の階層構造を示す図である。

【図 2 3】

構造化文書の構成と処理上の区分との関係を示す図である。

【図 2 4】

XML プロセッサの処理を示す図である。

【図 2 5】

XML 文書を示す図である。

【図 2 6】

DOM の木構造を示す図である。

【図 2 7】

DOM を用いたタグ検索処理のフローチャートである。

【図 2 8】

SAX を用いたタグ検索処理のフローチャートである。

【符号の説明】

- 1 1 XML 宣言
- 1 2 文書型定義
- 1 3 XML 実現値
- 2 1 <要素名>
- 2 2 </要素名>
- 2 3 <要素名/>
- 3 1 XML 文書
- 3 2 XML プロセッサ
- 3 3 木構造

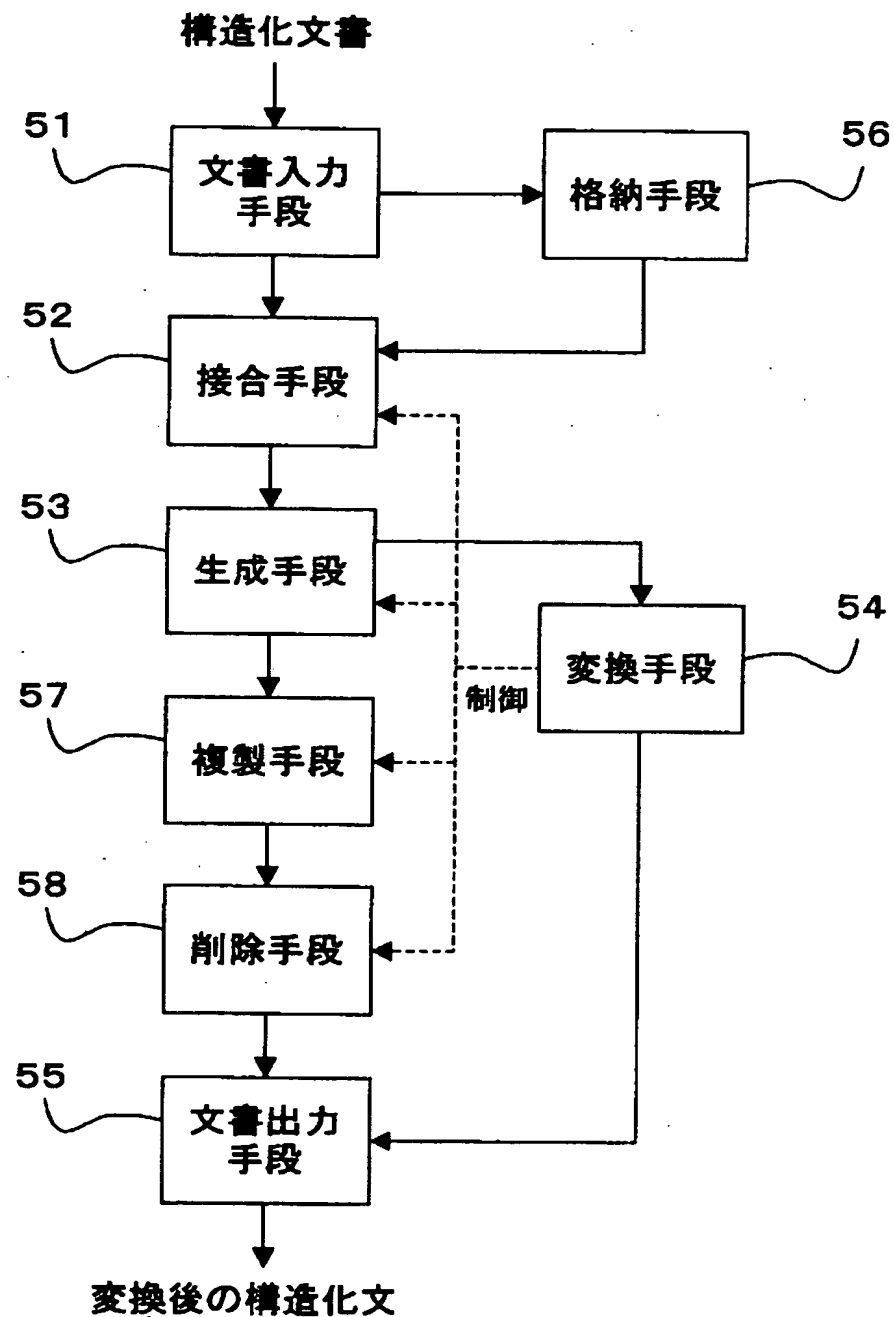
- 34 応用ソフトウェア
- 41 Document
- 42 NodeList
- 43 Element
- 44、75 Text
- 45 NamedNodeMap
- 46 Attr
- 51 文書入力手段
- 52 接合手段
- 53 生成手段
- 54 変換手段
- 55 文書出力手段
- 56 格納手段
- 57 複製手段
- 58 削除手段
- 61、71 EventList
- 62、72 Event
- 63、74 Info
- 73 Start
- 81 CPU
- 82 メモリ
- 85 外部記憶装置
- 86 媒体駆動装置
- 87 ネットワーク接続装置
- 88 バス
- 89 可搬記録媒体
- 90 サーバ
- 91 データベース

【書類名】

図面

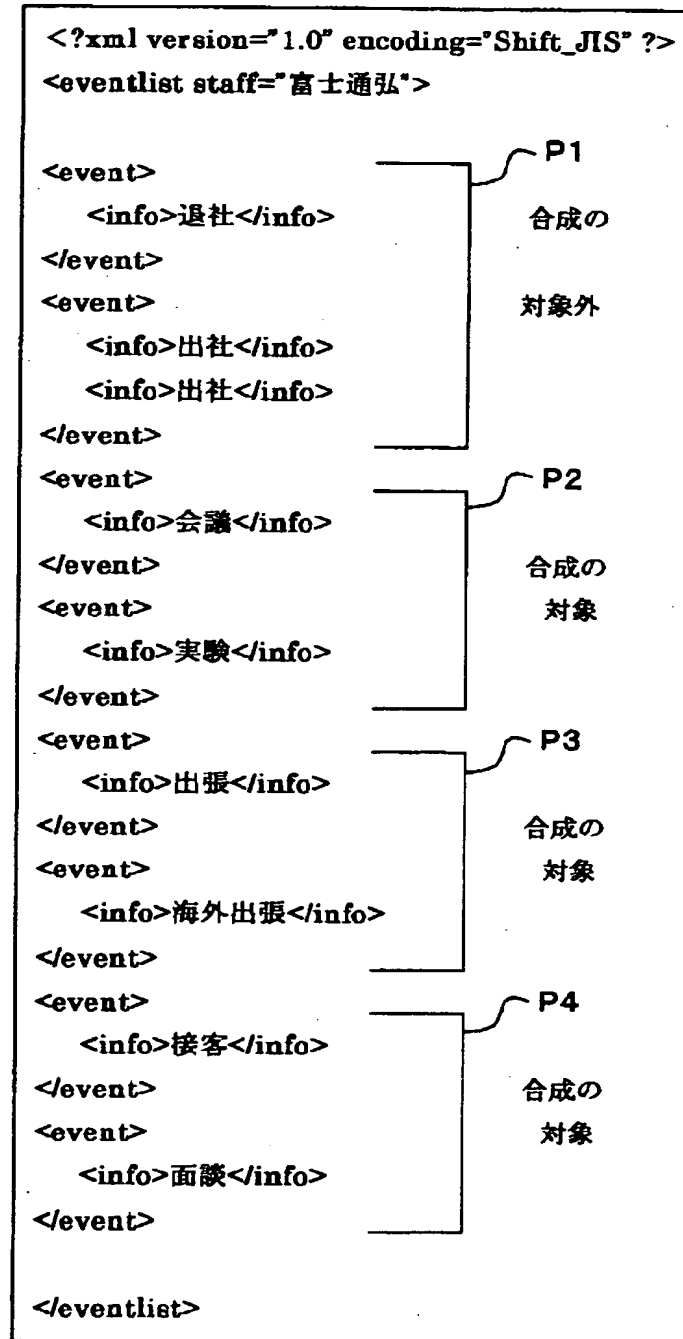
【図 1】

本発明の原理図



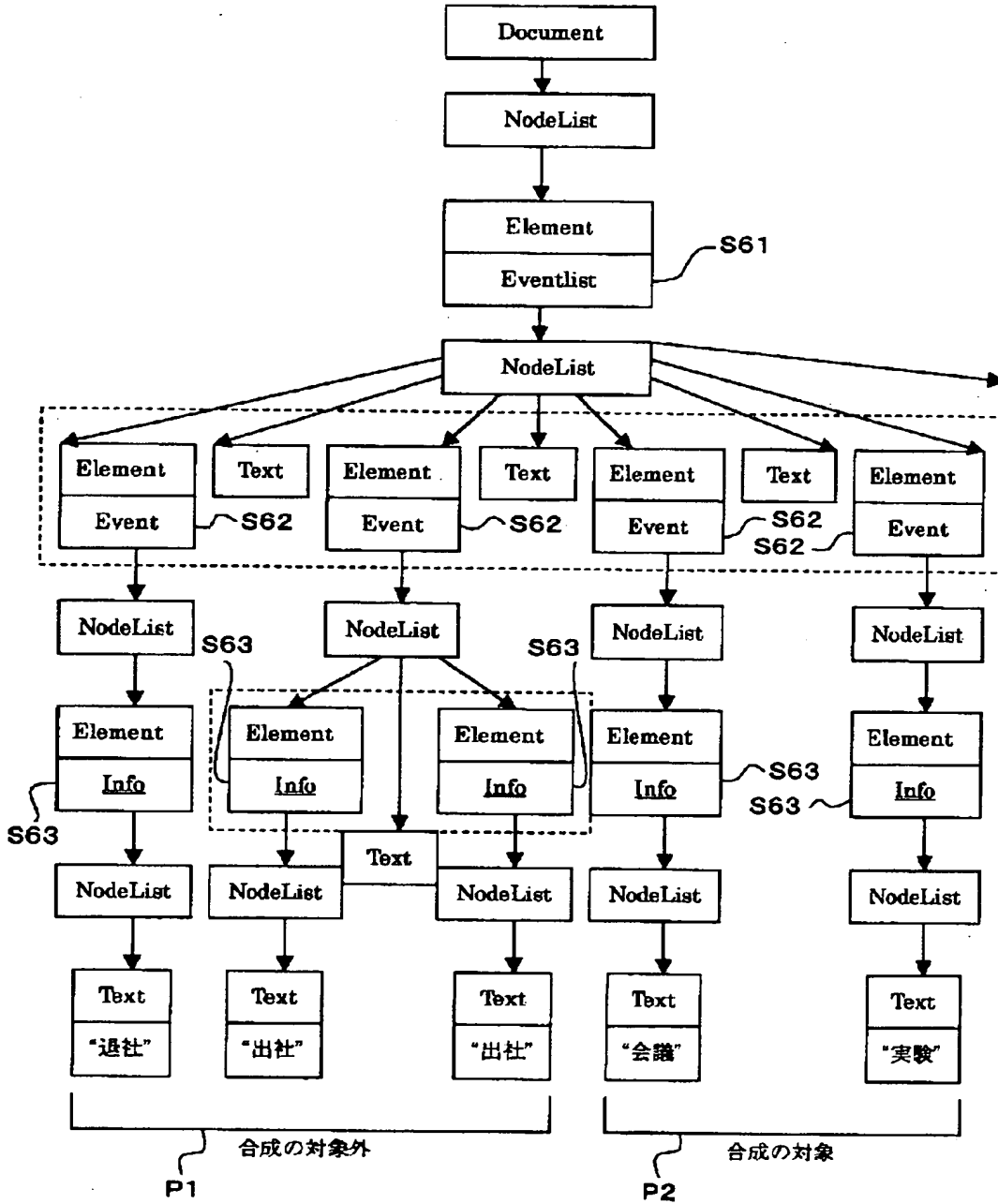
【図 2】

第1の処理対象のXML文書を示す図



【図 3】

第1の処理対象の木構造を示す図



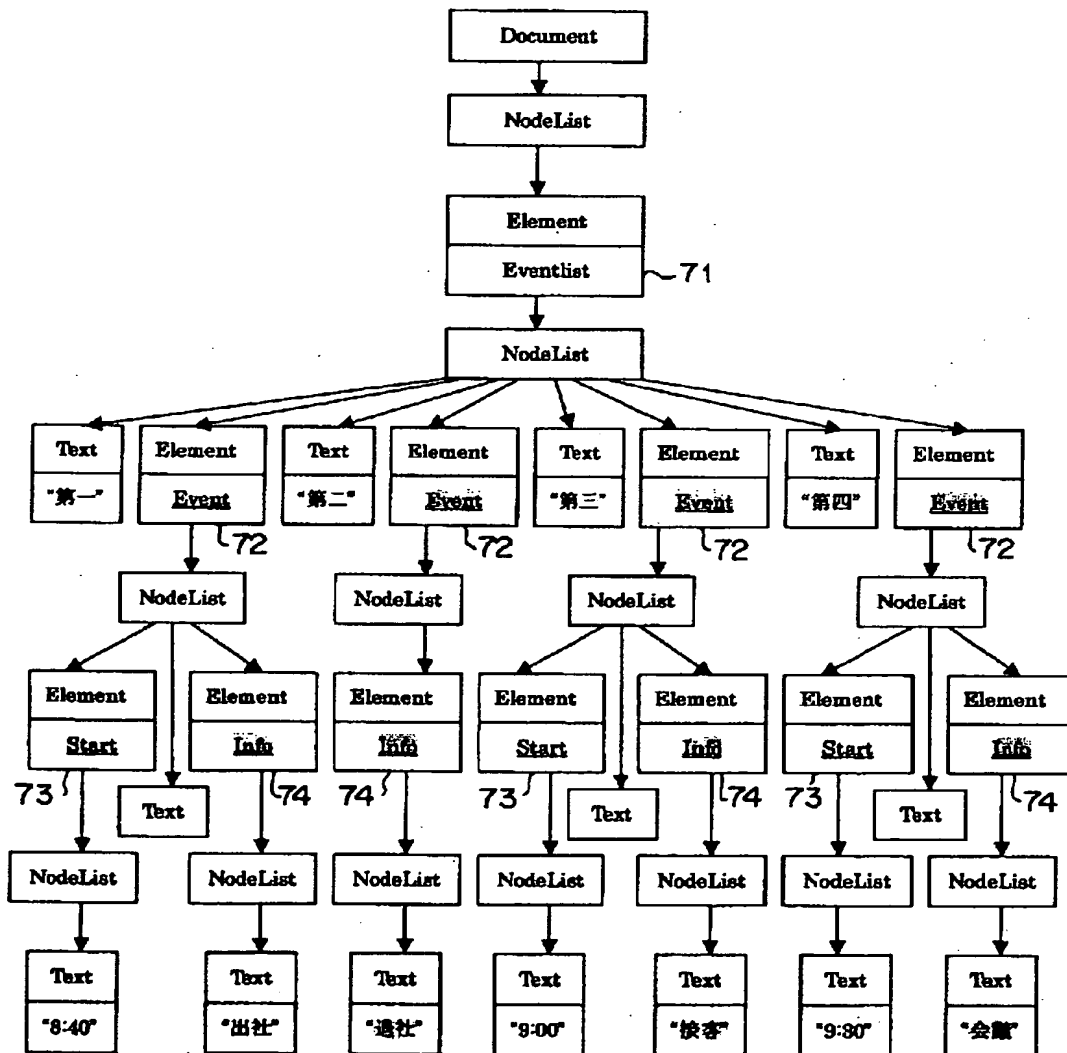
【図 4】

第2の処理対象のXML文書を示す図

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<eventlist staff="富士通弘">
  第一
  <event>
    <start>8:40</start>
    <info>出社</info>
  </event>
  第二
  <event>
    <info>退社</info>
  </event>
  第三
  <event>
    <start>9:00</start>
    <info>接客</info>
  </event>
  第四
  <event>
    <start>9:30</start>
    <info>会議</info>
  </event>
</eventlist>
```

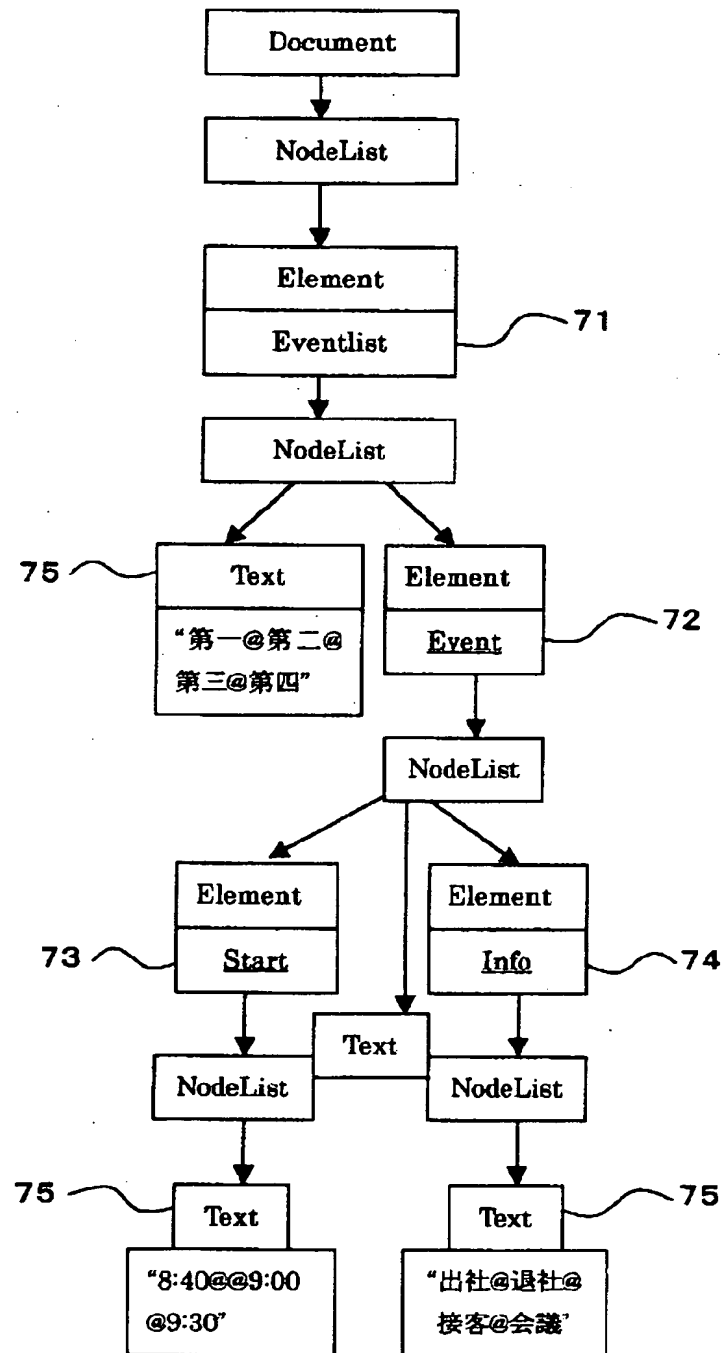
【図 5】

第2の処理対象の木構造を示す図



【図 6】

合成後の木構造を示す図



【図7】

第1の合成後のXML文書を示す図

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<eventlist staff="富士通弘">
  第一@第二@第三@第四
  <event>
    <start>8:40@@9:00@9:30</start>
    <info>出社@退社@接客@会議</info>
  </event>
</eventlist>
```

【図 8】

第1のハッシュ表を示す図

ハッシュ値	先祖、親の要素名	自身の要素名	要素内容
H 1	Event	Info	出社@退社@接客@会議
H 2	Event	Start	8:40@@9:00@9:30

【図 9】

第2のハッシュ表を示す図

ハッシュ値	先祖、親の要素名	自身の要素名	要素内容
H 3	Event	Info	出社@退社@接客@会議
H 4	Event	Start	8:40@17:00@9:00@9:30

【図10】

第3の処理対象のXML文書を示す図

```
<?XML version="1.0"?>
<DOCTYPE doc>
<doc>
  <個人>
    <姓>鈴木</姓>
    <名>小百合</名>
    <旧姓>小川</旧姓>
  </個人>
  <個人>
    <姓>佐藤</姓>
    <名>一郎</名>
  </個人>
</doc>
```

【図11】

第2の合成後のXML文書を示す図

```
<?XML version="1.0"?>
<DOCTYPE doc>
<doc>
<個人>
<姓>鈴木@佐藤</姓>
<名>小百合@一郎</名>
<旧姓>小川@</旧姓>
</個人>
</doc>
```

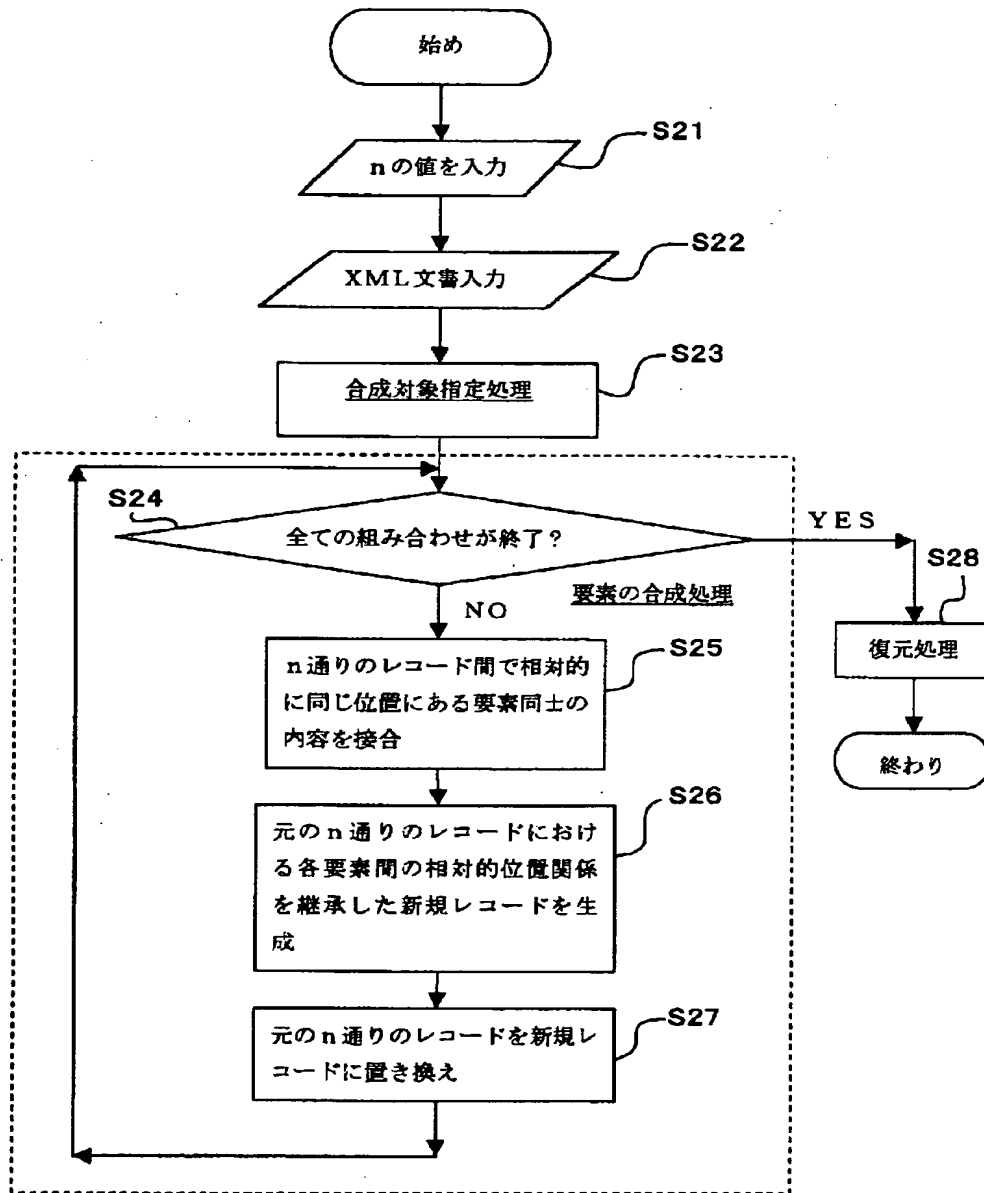
【図12】

検索結果を示す図

```
<?XML version="1.0"?>
<DOCTYPE doc>
<doc>
<個人>
<姓>鈴木</姓>
<名>小百合</名>
<旧姓>小川</旧姓>
</個人>
</doc>
```

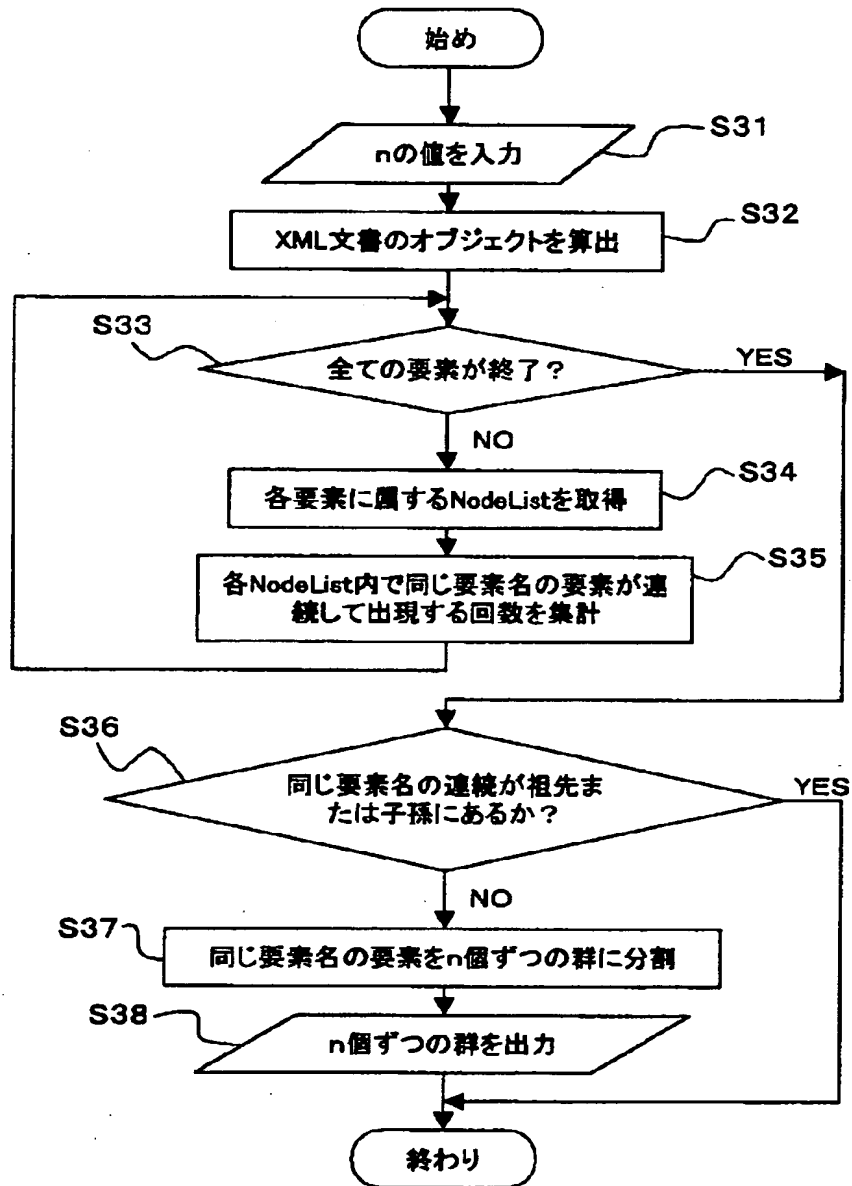
【図 13】

XML文書の変換を含むタグ検索処理のフローチャート



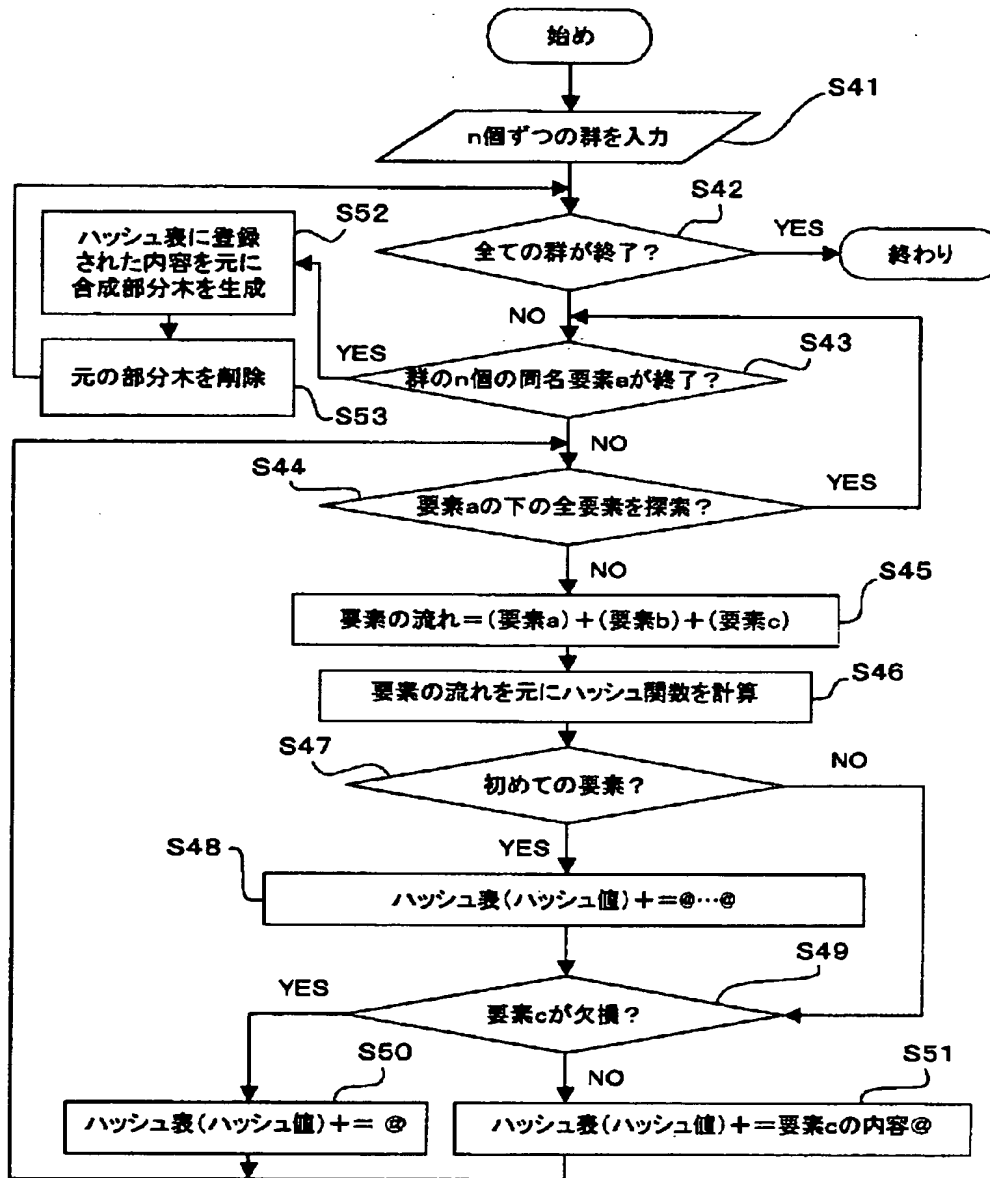
【図 14】

合成対象指定処理のフローチャート



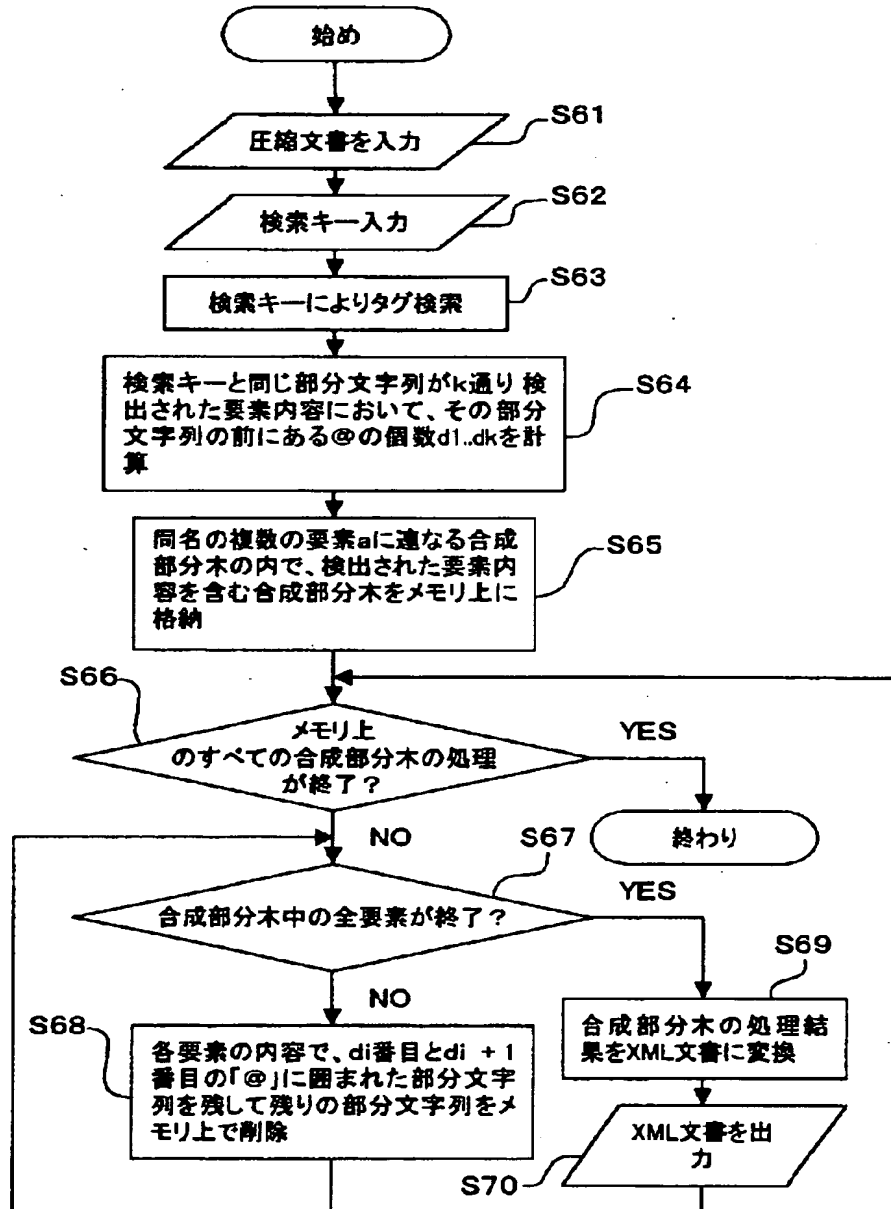
【図 15】

要素合成処理のフローチャート



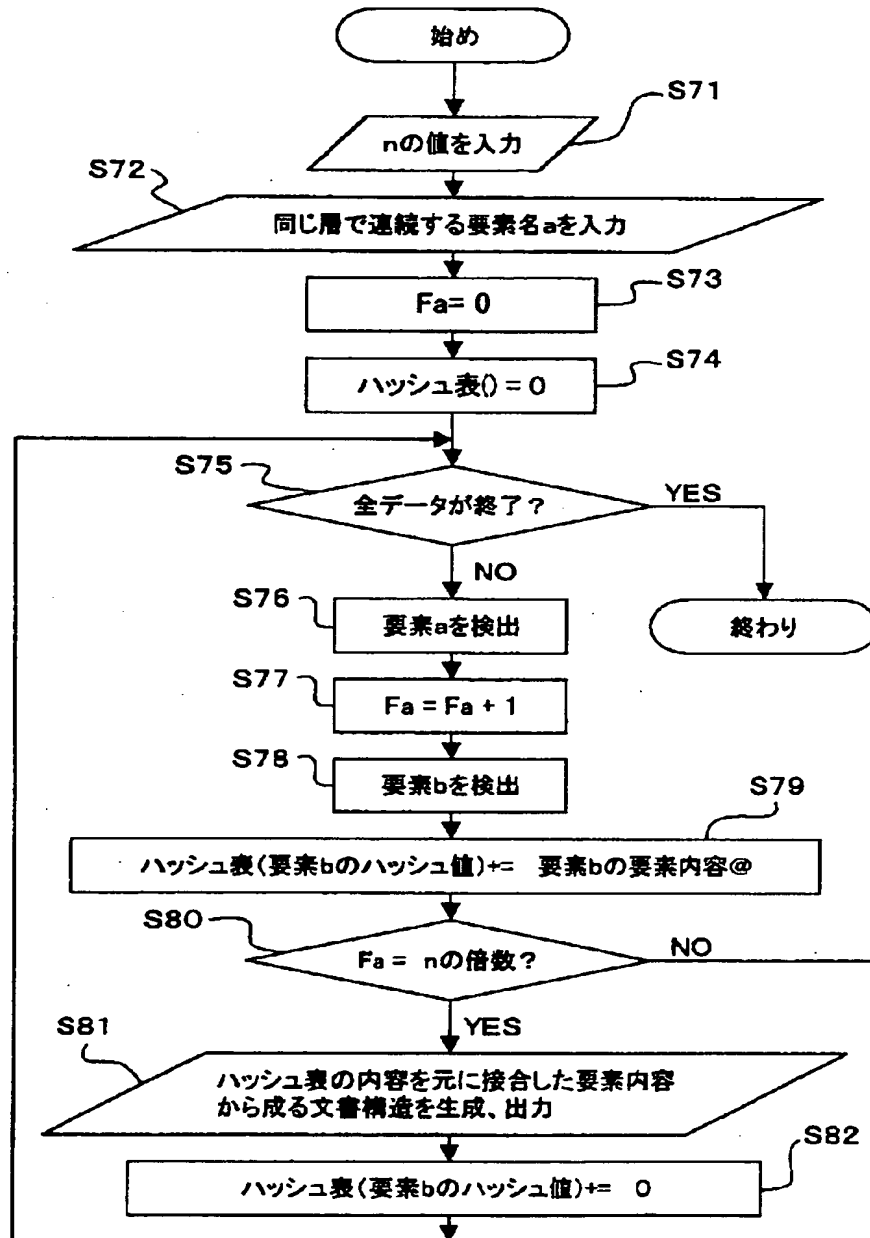
【図16】

復元処理のフローチャート



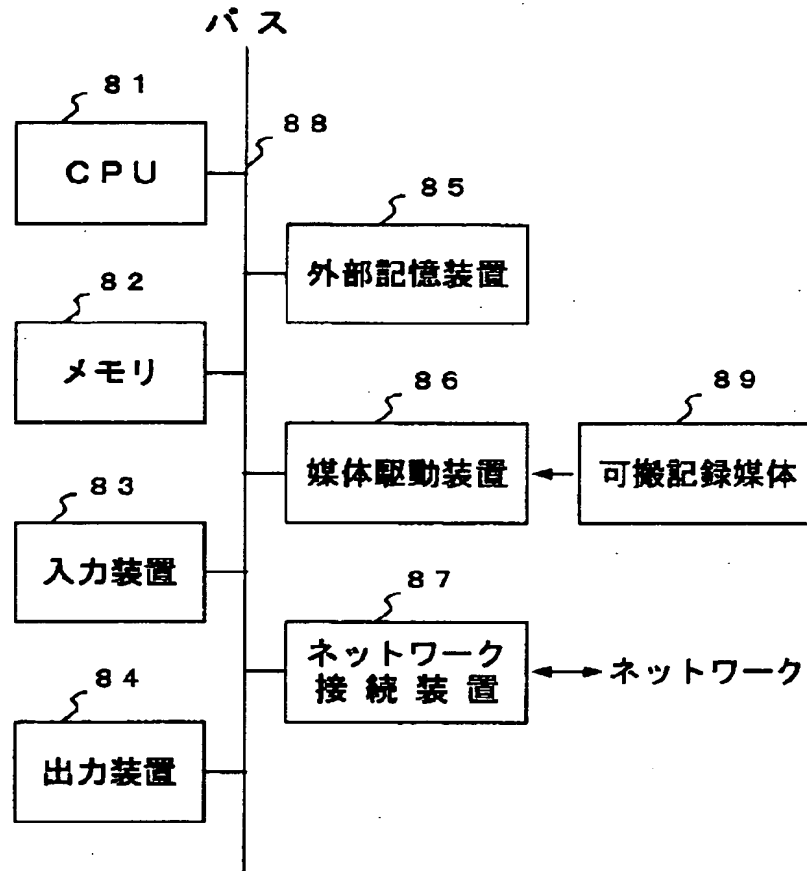
【図 17】

簡易型合成処理のフローチャート



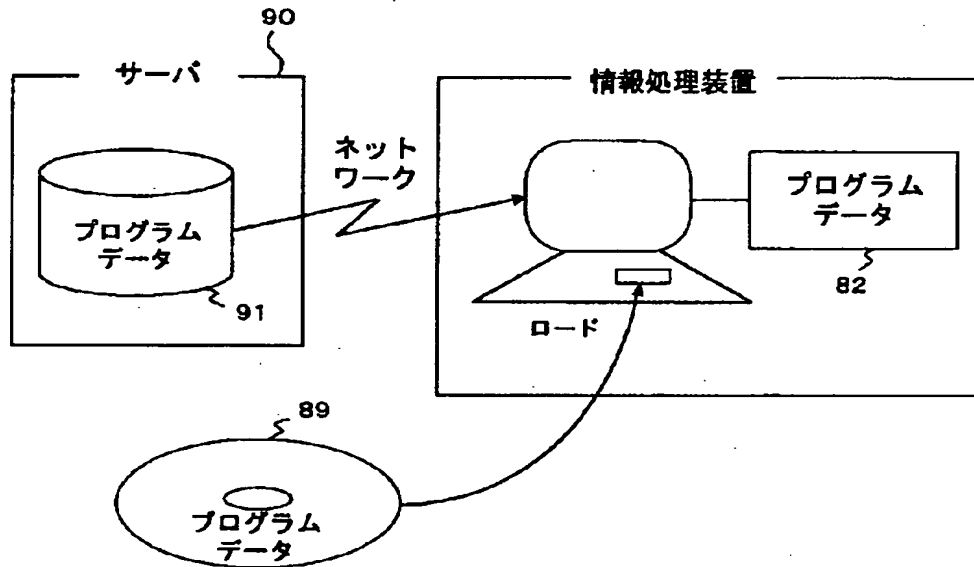
【図 18】

情 報 処 理 装 置 の 構 成 図



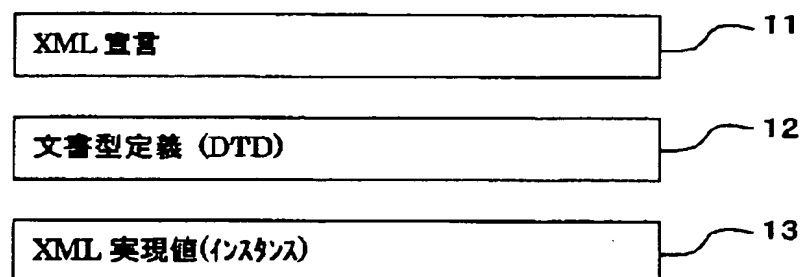
【図 1 9】

記 録 媒 体 を 示 す 図



【図 2 0】

XML文書の構成部分を示す図



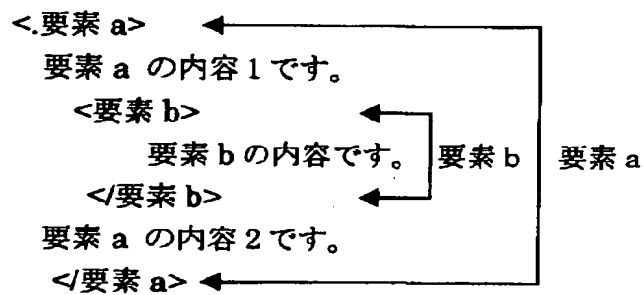
【図 2 1】

タグの書き方を示す図

- 21 ———— <要素名> (開始タグ)
 要素の内容です。(要素の内容)
- 22 ———— </要素名> (終了タグ)
- 23 ———— <要素名/> (空要素タグ：要素の内容のないタグ)

【図 2 2】

要素の階層構造を示す図



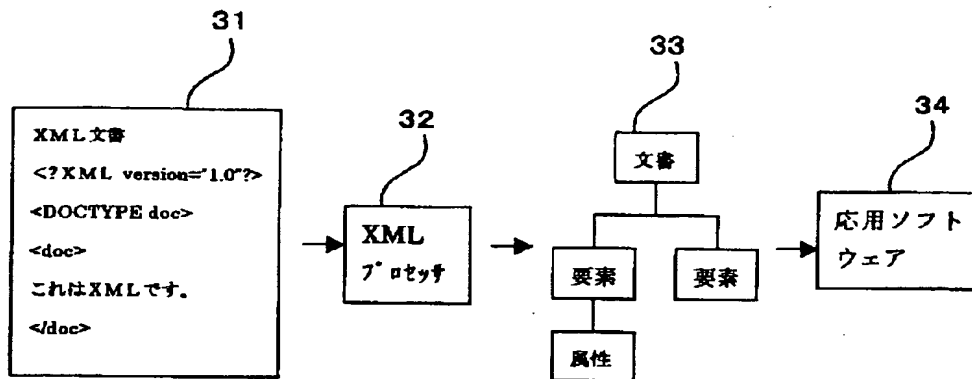
【図 23】

構造化文書の構成と処理上の区分との関係を示す図

	各種宣言	文書型定義	実現値
整形 XML 文書	△ (XML 宣言)	△	○
検証済み XML 文書	△ (XML 宣言)	○	○
SGML 文書	○ (SGML 宣言)	○	○
HTML 文書	△ (HTML 宣言)	○	○

【図 24】

XMLプロセッサの処理を示す図



【図 25】

XML文書を示す図

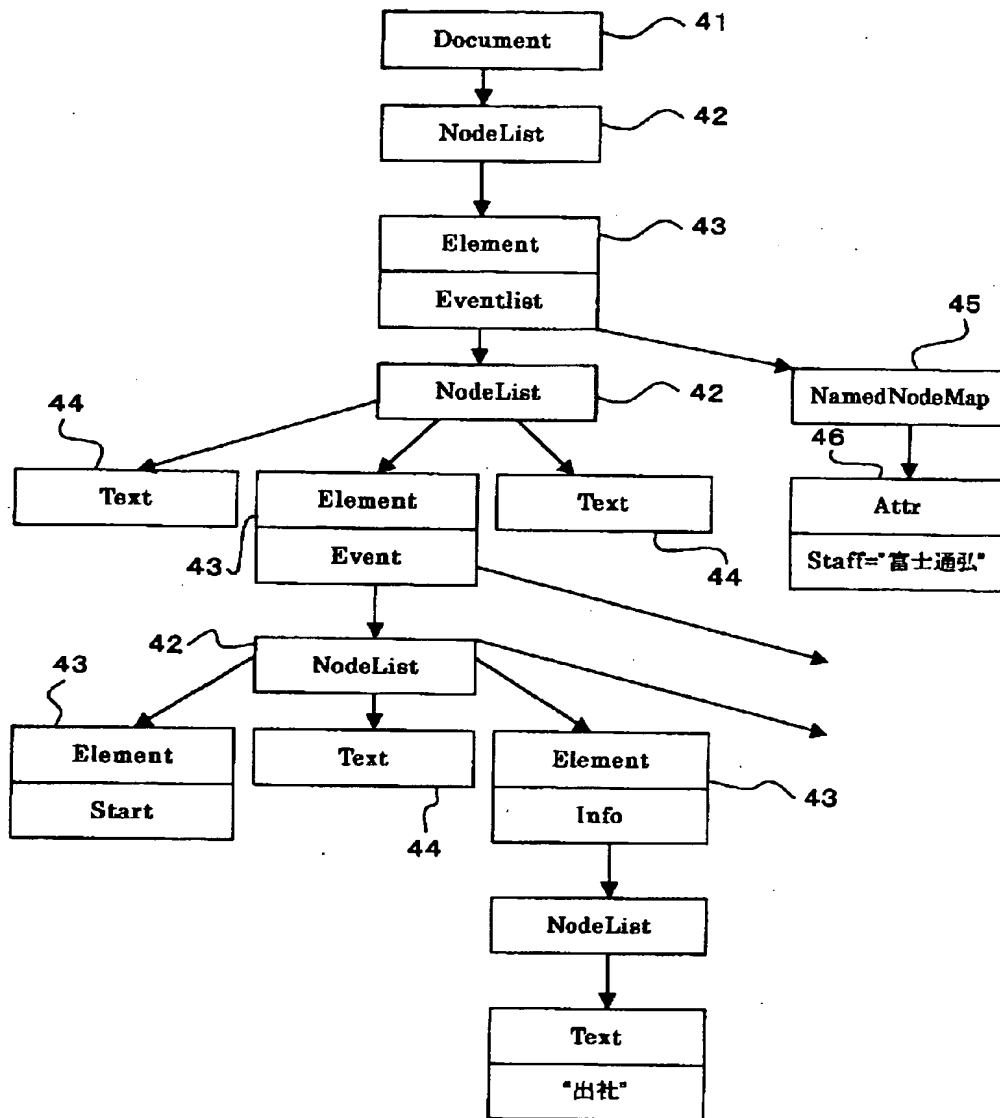
```

<?xml version="1.0" encoding="Shift_JIS"?>
<eventlist staff="富士通弘">
  <event type="業務">
    <start/>
    <info>出社</info>
  </event>
</eventlist>
    
```

This figure shows an example of an XML document. The root element is 'eventlist' with an attribute 'staff' set to '富士通弘'. It contains one child element 'event' with an attribute 'type' set to '業務'. The 'event' element has two children: 'start' (a self-closing tag) and 'info' (containing the text '出社').

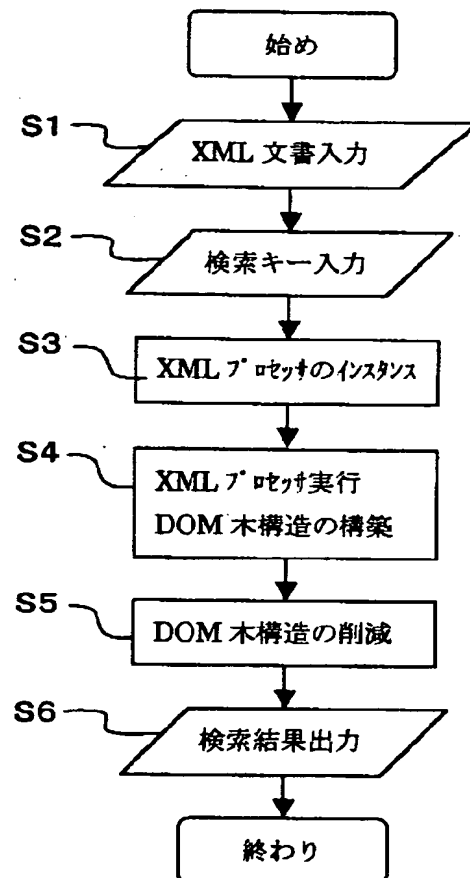
【図 26】

DOMの木構造を示す図



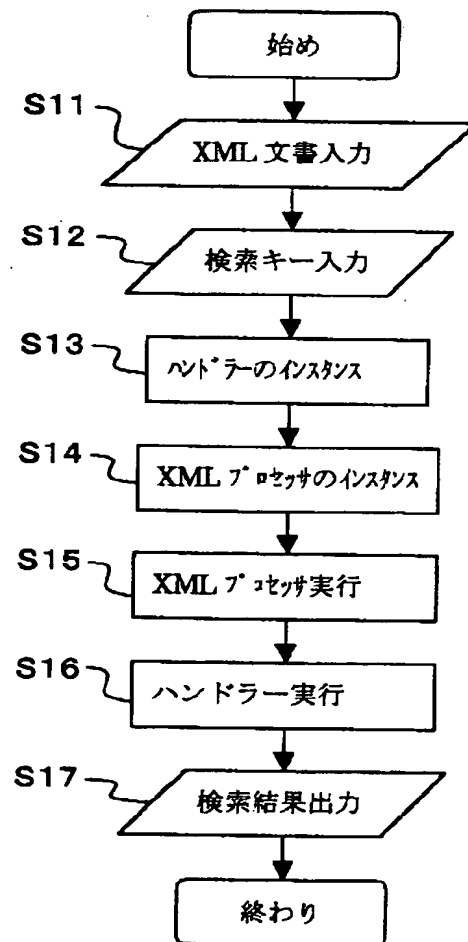
【図 2 7】

DOMを用いたタグ検索処理のフローチャート



【図 28】

SAXを用いたタグ検索処理のフローチャート



【書類名】 要約書

【要約】

【課題】 XML文書のような構造化文書を変換することにより、タグ検索の処理速度を向上させ、必要な動作メモリ量を削減することが課題である。

【解決手段】 構造化文書を構成するP2、P3、およびP4の各部分に含まれる複数のレコードを合成対象として、それらのレコード間で同じ位置にある要素の内容を接合し、接合された内容を持つ新しい要素を生成する。そして、新しい要素を用いて、元のレコードにおける要素の相対的位置関係を継承した新しいレコードを生成し、元のレコードを新しいレコードに置き換えて、文書を変換する。

【選択図】 図2

出 願 人 履 歴 情 報

識別番号 [000005223]

1. 変更年月日	1996年 3月26日
[変更理由]	住所変更
住 所	神奈川県川崎市中原区上小田中4丁目1番1号
氏 名	富士通株式会社